1991年8月1日発行（毎月1回1日発行）第10巻8号通巻112号　昭和58年11月2日第三種郵便物認可

PERSONAL COMPUTER MAGAZINE for MZ, X1, and X68000

特集 印刷の世界へ

製品紹介　イメージジェットプリンタIO-735X-B/イメージスキャナJX-220X
音源モジュールSOUND CANVAS SC-55/3.5"FDD TS-3XR1
SX-WINDOW対応グラフィックツールEasypaintSX-68K

CARDDRV用ゲーム「七並べ」/X1用シューティングゲームDEFEAT2

1991

オー！エックス
定価600円

SHARP
仕事だけのパソコンや
ワープロみたいなパソコンは、
いらない。
父のパソコンを超えろ。

いまクロック16MHzの俊才、「エクシヴィ」のデビューで5年に及ぶ68000CPUへの探求は、ひとつの結論を得ようとしています。極めたといえば言い過ぎでしょうが、事の深淵に迫ろうと努力するもののみに与えられる深い充足を、私たちスタッフは、これまでX68000を支えていただいたユーザー、ソフトハウス、ハードベンダー諸兄とともに味わいたい心境です。徹底したこだわりと、それを裏付けるアドバンストテクノロジー、世間の逆風を揚力にしてしまう、それなりの魅力と知性を背景として備えたX68000が、パーソナルコンピュータに新しいジャンルを切り拓いてきた歩みは、ご存じの通りです。現在のマルチメディア環境を開発当初から想定していた先見性。一言でいえばクリエイティブマインドということでしょうが、そのグラフィックアビリティ、映像統合コンセプト、サンプリング音源、ウィンドウ環境、そうした単に、とはいえ凄いスペックさえ超えたところにX68000の付加価値は存在します。アプリケーションを走らせるだけのブラックボックス化した、あるいは文房具としてのマシン、それはそれで異論はないのですが、本来的にパーソナルコンピュータがもつ可能性を育む、いわば創造性という観点から物足りなさを覚えることも事実です。X68000は、ある意味ではたいへんな異端児かも知れません。しかし世間から見たその"異能"は、私たちが考えるパーソナルコンピュータとしてはまさにスタンダードに他なりません。いつも新鮮な感動がある、驚きがある。新しい発見がある。"センス"の違いはスペックをも超えて使う人に訴えかける、敢えて68000CPUに執着してきた理由もここにあります。ワークステーションとしての成熟、先見性、創造性の具現化、ユーザーインターフェイスの追求。X68000の進化の過程はここに凝縮されています。

——新しい「エクシヴィ」がこのコンセプトをどう発展させたか、ご体感ください。

瞬速16MHz、エクシヴィ登場。

16MHzクロック68000搭載：OSの高速化、アートワークをパワフルにサポートするクロック周波数16MHzの68000CPUを搭載。クリエイティブワークステーションにふさわしいシステムパフォーマンスを実現しました。

SX-WINDOW ver1.1搭載：CPUのクロックアップと合わせ、大幅な処理速度の向上を実現。操作性を一段と高めたニューバージョンです。多機能・高速の強力エディタを搭載。文字選択・外字作成ツールも装備して、スムーズな日本語入力環境をサポート。またプリンタデバイスドライバを搭載し、多彩な印字指定が可能です。もちろん、こうした新しい環境がすべてのX68000で享受できることは言うまでもありません。そして待望のウィンドウアプリケーションもリリースされはじめています。

高密度メモリ拡張環境：メインメモリは標準で2Mバイト、本体内部のメイン基板上に6Mバイト増設でき、I/Oスロットを使用せず最大8Mバイトの高速メモリアクセスを実現。さらにI/Oスロットへの増設を含め最大12Mバイトまで拡張できます。数値演算プロセッサも本体内に取り付けられます。

※2MB増設メモリ（ボード型）CZ-6BE2A　標準価格59,800円（税別）、2MB増設メモリ（チップ型）CZ-6BE2B　標準価格54,800円（税別）、数値演算プロセッサ（チップ型）CZ-6BP2　標準価格45,800円（税別）を使用。（すべて別売）

●大容量メディア対応、世界標準SCSIインターフェイス標準装備●X68000シリーズとフルコンパチブル設計●高品位なチタンブラックのニューデザインマンハッタンシェイプ●81MバイトSCSI仕様HDD搭載（CZ-644C）/内蔵可能（CZ-634C）●1024×1024ドットの実画面エリアを装備した高解像度表示（最大表示エリア768×512ドット・65,536色中16色表示）、65,536色同時表示（512×512ドット時）、先駆の高解像度自然色グラフィック●AD PCM、ステレオ8オクターブ8重和音FM音源搭載●オートロード・オートイジェクトの1Mバイト5インチFDD2基搭載●マウス・トラックボール標準装備

●写真はCZ-644C-TNとCZ-613D-TN。

本体＋キーボード＋マウス・トラックボール
CZ-634C-TN（チタンブラック）　標準価格368,000円（税別）
81MB HDタイプ　CZ-644C-TN（チタンブラック）　標準価格518,000円（税別）

SUPER 本体＋キーボード＋マウス・トラックボール
CZ-604C-TN（チタンブラック）　標準価格348,000円（税別）
81MB HDタイプCZ-623C-TN（チタンブラック）　標準価格498,000円（税別）

PROII 本体＋キーボード＋マウス
CZ-653C-BK（ブラック）・-GY（グレー）　標準価格285,000円（税別）
40MB HDタイプCZ-663C-BK（ブラック）・-GY（グレー）　標準価格395,000円（税別）

●15型カラーディスプレイテレビ（ドットピッチ0.39mm）　CZ-602D-BK（ブラック）・-GY（グレー※）　標準価格99,800円（チルトスタンド同梱・税別）
●15型カラーディスプレイテレビ（ドットピッチ0.39mm）　CZ-605D-BK（ブラック）・-GY（グレー）　標準価格115,000円（スピーカー2個/チルトスタンド同梱・税別）
●15型カラーディスプレイテレビ（ドットピッチ0.31mm）　CZ-613D-TN（チタンブラック）・-BK（ブラック）・-GY（グレー）　標準価格135,000円（スピーカー2個/チルトスタンド同梱・税別）
●14型カラーディスプレイ（ドットピッチ0.31mm）　CZ-603D-BK（ブラック）・-GY（グレー）　標準価格84,800円（チルトスタンド同梱・税別）
●14型カラーディスプレイ（ドットピッチ0.31mm）　CZ-604D-BK（ブラック）・-GY（グレー）　標準価格94,800円（スピーカー2個/チルトスタンド同梱・税別）
●14型カラーディスプレイ（ドットピッチ0.31mm）　CZ-606D-TN（チタンブラック）・-BK（ブラック）・-GY（グレー）　標準価格79,800円（チルトスタンド同梱・税別）
●21型カラーディスプレイ（ドットピッチ0.52mm）　CU-21HD-BK（ブラック）　標準価格148,000円（スピーカー2個同梱・税別）

※印の商品は在庫僅少です。

いよいよ予選大会開始!! お友達を連れてぜひ、ご来場下さい。

開催日	開催地	会場	応募・問い合わせ先
7月21日(日)	四国大会	シャープ高松ビル 5Fイベントホール 高松市朝日町6-2-8　☎0878-23-4860	シャープエレクトロニクス販売㈱ 四国統轄(営)パソコン担当　☎0878-23-4860㈹
8月4日(日)	北海道大会	シャープ札幌ビル 4Fホール 札幌市西区二十四軒1条7-3-17　☎011-642-8111	シャープエレクトロニクス販売㈱ 北海道統轄(営)パソコン担当☎011-642-8111㈹

●お問い合わせは…
シャープ株式会社
電子機器事業本部システム機器営業部
〒545 大阪市阿倍野区長池町22番22号☎(06)621-1221(大代表)
電子機器事業本部液晶映像システム事業部第2商品企画部
〒162 東京都新宿区市谷八幡町8番地☎(03)3260-1161(大代表)

Oh!X

C O N T

特集 印刷の世界へ

カラーイメージジェットプリンタ IO-735X

DEFEAT 2

黄金の羅針盤

サイレントメビウス

TS-3XRI

●特集

●カラー紹介

●THE SOFTOUCH

〈スタッフ〉
●編集長／前田 徹 ●副編集長／植木章夫 ●編集／岡崎栄子 浅井研二 山田純二 ●協力／有田隆也 中森 章 林 一樹 吉田幸一 華門真人 毛内俊行 吉田賢司 影山裕昭 古村 聡 村田敏幸 丹 明彦 三沢和彦 長沢淳博 宮島 靖 金子俊一 浦川博之 石上達也 ●カメラ／杉山和美 ●イラスト／永沢しげる 山田晴久 小栗由香 ●アートディレクター／島村勝頼 ●レイアウト／元木昌子 ADGREEN ●校正／グループこじら

表紙絵：須藤　牧人

1991 AUG.

8

E　N　T　S

■広告目次

SHARP

ウィンドウアプリケーション登場!

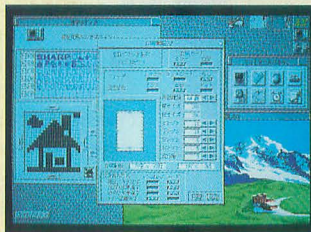
SX-WORKS

● 次代のウィンドウ環境を提供

SX-WINDOW ver1.1

CZ-278SS
標準価格9,800円(税別)

表示の高速化、強力テキストエディタを標準装備。またプリンタドライバを一新するなど、さらに操作環境が強化されたウィンドウシステムです。

※メインメモリ2MBが必要です。
※SX-WINDOW ver1.0(CZ-259SS)をお持ちの方、またX68000SUPER(CZ-623C、604C)・EXPERT II(CZ-603C、613C)・PRO II(CZ-653C、663C)に同梱のSX-WINDOW ver1.0をお持ちの方には有償バージョンアップサービスを行います。

● SX-WINDOW対応ペイントツール

Easypaint SX-68K

CZ-263GW
標準価格12,800円(税別)

使いやすさを追求したSX-WINDOW対応初のペイントツールです。

※メインメモリ2MBおよびSX-WINDOW ver1.1が必要です。

■Easydraw SX-68K(開発中)
SX-WINDOW上のドローソフト。初心者にも容易な操作性でオブジェクト図形が簡単に作成できます。

■Communication SX-68K(開発中)
SX-WINDOWの特長を生かした通信ソフト。簡単に通信ができるやさしいユーザーインターフェイスを持っています。

■SOUND SX-68K(開発中)
SX-WINDOW対応FM音源音色作成サウンドツールです。
自動演奏モニタウィンドウ付きで手軽に音色作成ができます。

資料請求券
CZソフト
Oh!X
8条

MONTHLY PICK UP

●シューティングゲーム

中華大仙

CZ-268AS　標準価格7,900円(税別)

©TAITO CORP. 1988

●コミカルアクションゲーム

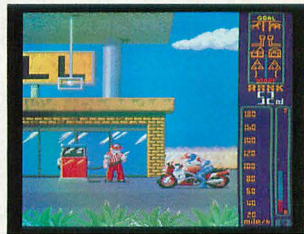
ボナンザブラザース

CZ-270AS　開発中

©SEGA1990 REPROGRAMMED BY SHARP

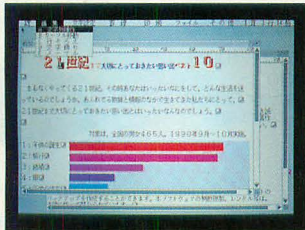
●バイクレーシングゲーム

ダッシュ野郎

CZ-269AS　標準価格8,800円(税別)

©TOAPLAN Co. Ltd. 1988

●ウィンドウでWYSIWYG編集

マルチワープロ PRO-68K Multiword

CZ-225BS 標準価格32,000円(税別)

マルチウィンドウを駆使したウィンドウモードと、素早い編集が可能な高速テキストモードを装備。カラーグラフィックも自由にレイアウト。レーザープリンタにも対応したマルチワープロです。

※メインメモリ2MB必要です。

●Zeit日本語ベクトルフォントをサポート

CZ-265HS 標準価格20,000円(税別)

処理速度の高速化はもちろん、カセットレーベル、カレンダー作成に対応したほか、モノクロデータの編集などグラフィックエディタを強化した高機能テキストエディタを内蔵しています。

※メインメモリ2MB必要です。
※NEW Print Shop PRO-68K(CZ-221HS)をお持ちの方には有償バージョンアップを行います。

●メモリ常駐型の優れモノ

Teleportion PRO-68K

CZ-258BS 標準価格22,800円(税別)

他のソフトを実行中でも任意に呼び出して使えるメモリ常駐型のソフト。パソコン通信/エディタ/スケジュール/住所録/メモ帳などの機能を文具感覚で使えます。

※メインメモリ2MB必要です。
※Stationery PRO-68K(CZ-240BS)をお持ちの方には有償バージョンアップを行います。

●高速カード型リレーショナルデータベース

CZ-253BS 標準価格29,800円(税別)

操作性の向上、高速化を図った新マルチウィンドウシステムを搭載したニューバージョンです。一覧表画面入力、グラフ機能などをサポート。キーボード操作にも対応します。

※メインメモリ2MB必要です。
※CARD PRO-68K(CZ-226BS)をお持ちの方には有償バージョンアップを行います。

《お詫びと訂正》■弊社発行のX68000ソフト情報誌「ソフトウェアフィールド」20号において、一部標準価格に誤りがありますので訂正させていただくとともに、謹んでお詫び申し上げます。

●Musicstudio PRO-68K ver2.0(CZ-261MS)……(誤)標準価格 18,800円(税別)→(正)標準価格 28,800円(税別)
●中華大仙(CZ-268AS)……(誤)標準価格 8,800円(税別)→(正)標準価格 7,900円(税別)
●光磁気ディスクユニット(CZ-6MO1)……(誤)標準価格 29,800円(税別)→(正)標準価格450,000円(税別)
●SCSIボード(CZ-6BS1)……(誤)標準価格450,000円(税別)→(正)標準価格 29,800円(税別)

※CZ-258BS、CZ-265HS、CZ-253BS、CZ-278SSの有償バージョンアップについては、下記にお問い合わせください。

●お問い合わせは…シャープ㈱電子機器事業本部液晶映像システム事業部第2商品企画部 〒162 東京都新宿区市谷八幡町8番地 ☎(03)3260-1161(大代表)へ。シャープ株式会社

SHARP

システムパフォーマンスを実証する多彩なペリフェラル。

ディスプレイ関連

カラーディスプレイテレビ

14型カラーディスプレイテレビ
CZ-607D-BK・-TN
標準価格99,800円(税別)
(チルトスタンド同梱)

15型カラーディスプレイテレビ
CZ-605D-BK・-GY
標準価格115,000円(税別)
(スピーカー2個・チルトスタンド同梱)

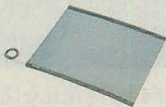
15型カラーディスプレイテレビ
CZ-614D-BK・-TN
標準価格135,000円(税別)
(スピーカー2個・チルトスタンド同梱)

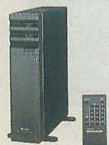
カラーディスプレイ

14型カラーディスプレイ
CZ-606D-TN・BK・-GY
標準価格79,800円(税別)
(チルトスタンド同梱)

14型カラーディスプレイ
CZ-604D-BK・-GY
標準価格94,800円(税別)
(スピーカー2個・チルトスタンド同梱)

21型カラーディスプレイ
CU-21HD
標準価格148,000円(税別)
(スピーカー2個同梱)

CRTフィルター

高性能CRTフィルター
BF-68PRO
標準価格19,800円(税別)
(14/15型用)

チューナー

RGBシステムチューナー
CZ-6TU-BK・-GY
標準価格33,100円(税別)
(リモコン付)

アートツール

画像入力

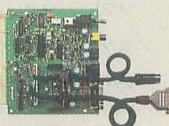
カラーイメージスキャナ※1
CZ-8NS1
標準価格188,000円(税別)

カラーイメージスキャナ※1
JX-220X
標準価格168,000円(税別)
※RS-232C/パラレルインターフェイス標準装備

スキャナ用パラレルボード
CZ-6BN1
標準価格29,800円(税別)

映像入力

カラーイメージユニット※2
CZ-6VT1-BK
CZ-6VT1
標準価格69,800円(税別)

映像出力

ビデオボード※3
CZ-6BV1
標準価格21,000円(税別)

プリンタ

熱転写カラープリンタ

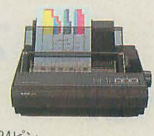
48ドット
熱転写カラー漢字プリンタ
CZ-8PC5-BK
標準価格96,800円(税別)

カラービデオプリンタ

カラービデオプリンタ
★CZ-6PV1
標準価格198,000円(税別)
(信号ケーブル同梱)

カラーイメージジェット

カラーイメージジェット※4
IO-735X-B
標準価格248,000円(税別)
(信号ケーブル別売)
※グレータイプのIO-735Xもあります。

カラードットプリンタ

24ピン
カラー漢字プリンタ(80桁)
CZ-8PG1
標準価格130,000円(税別)
(信号ケーブル同梱)

24ピン
カラー漢字プリンタ(136桁)
CZ-8PG2
標準価格160,000円(税別)
(信号ケーブル同梱)

ドットプリンタ

24ピン漢字プリンタ(136桁)
CZ-8PK10
標準価格97,800円(税別)
(信号ケーブル同梱)

ファイル

光磁気ディスク

光磁気ディスクユニット※5
(594MB)
CZ-6MO1
標準価格450,000円(税別)
(SCSIケーブル同梱)
※光磁気ディスクカートリッジは別売です。別売のJY-701MPA 標準価格30,000円(税別)をご使用ください。

ハードディスク

増設用ハードディスクドライブ(40MB)
(CZ-602C/603C/652C/653C内蔵用)
★CZ-64H※
標準価格120,000円(税別)
(取付費別)

増設用ハードディスクドライブ(81MB)
(CZ-604C/634C内蔵用)
CZ-68H※
標準価格160,000円(税別)
(取付費別)
※取付に関してはシャープお客様ご相談窓口にてご相談ください。

ハードディスクユニット(20MB)
★CZ-620H
標準価格178,000円(税別)
※604C/623C/634C/644Cでは使用できません。

※1 ご使用に際しては、カラーイメージスキャナCZ-8NS1、JX-220Xに同梱のRS-232Cケーブルで接続するか、より高速のパラレルデータ伝送を行う場合、別売のスキャナ用パラレルボードCZ-6BN1標準価格29,800円(税別)で接続してください。※2 テレビチューナーを内蔵していないディスプレイをご使用の場合は、RGBシステムチューナーCZ-6TU(別売)が必要です。※3 ビデオ出力は15.75kHzテレビ標準信号です。また、拡張I/Oスロットは2スロット使用します。※4 別売の信号ケーブルIO-73CX 標準価格5,500円(税別)で接続してください。※5 CZ-600C、601C、602C、603C、611C、612C、613C、652C、653C、662C、663Cにご使用の場合は、別売のSCSIボード(CZ-6BS1)が必要です。また、X68000用 OS Human 68k ver 2.0以上にてご使用ください。(光磁気ディスクカートリッジは別売のJY-701MPA 標準価格30,000円(税別)をご使用ください。) ※6 ご使用に際しては、あらかじめ別売の1MB増設RAMボード CZ-6BE1 標準価格 35,000円(税別・

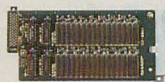
シャープ株式会社 ●お問い合わせは…シャープ(株)電子機器事業本部システム機器営業部 〒545 大阪市阿倍野区長池町22番22号☎(06)621-1221(大代表

お望みのパワーシステムへ。

ボード

拡張メモリ

2MB増設RAMボード
(CZ-634C/644C専用)
CZ-6BE2A
標準価格59,800円(税別)
※2MB増設RAM(CZ-6BE2B)専用ソケットを2個用意しています。

2MB増設RAM
(CZ-634C/644C専用)
CZ-6BE2B
標準価格54,800円(税別)
※本増設RAM(CZ-6BE2B)は、2MB増設RAMボードが必要です。CZ-6BE2A上の専用ソケット(2個用意)に装着ください。
※取付に関してはシャープお客様ご相談窓口にてご相談ください。

1MB増設RAMボード
(CZ-600C専用)
★CZ-6BE1
標準価格35,000円(税別)

1MB増設RAMボード
(CZ-601C/611C/652C/653C/662C/663C用)
CZ-6BE1B
標準価格28,000円(税別)

2MB増設RAMボード※6
CZ-6BE2
標準価格79,800円(税別)

4MB増設RAMボード※6
CZ-6BE4
標準価格138,000円(税別)

インターフェイス

SCSIボード※7
CZ-6BS1
標準価格29,800円(税別)
(ソフトウェア(SCSIユーティリティ)同梱)

ユニバーサルI/Oボード
★CZ-6BU1
標準価格39,800円(税別)

GP-IBボード
★CZ-6BG1
標準価格59,800円(税別)

増設用RS-232Cボード
(2チャンネル)
★CZ-6BF1
標準価格49,800円(税別)

MIDI

MIDIボード
CZ-6BM1
標準価格26,800円(税別)

FAX

FAXボード
CZ-6BC1
標準価格79,800円(税別)

数値演算プロセッサ

数値演算プロセッサボード
CZ-6BP1
標準価格79,800円(税別)

数値演算プロセッサ
(CZ-634C/644C専用)
CZ-6BP2
標準価格45,800円(税別)
※取付に関してはシャープお客様ご相談窓口にてご相談ください。
※特別ケース入りです。

ネットワーク

モデム

モデムユニット※8
CZ-8TM2
標準価格49,800円(税別)
(RS-232Cケーブル同梱)

RS-232Cケーブル

RS-232Cケーブル
(平行接続型)
CZ-8LM1
標準価格7,200円(税別)

RS-232Cケーブル
(クロス接続型)
CZ-8LM2
標準価格7,200円(税別)

LANボード

LANボード
CZ-6BL1
標準価格268,000円(税別)
(イーサネット用)

CZ-6BL2
標準価格298,000円(税別)
(イーサネット/チーパネット両用)
※電源ユニット・ソフトウェア(ネットワークドライバVer1.0)同梱

入力

インテリジェントコントローラ
CZ-8NJ2
標準価格23,800円(税別)

マウス・トラックボール
CZ-8NM3
標準価格9,800円(税別)

トラックボール
CZ-8NT1
標準価格13,800円(税別)

マウス
CZ-8NM2A
標準価格6,800円(税別)

ジョイカード
CZ-8NJ1
標準価格1,700円(税別)

その他

拡張スロット

拡張I/Oボックス(4スロット)
(CZ-600C/601C/602C/603C/604C/611C/612C/613C/623C/634C/644C用)
CZ-6EB1-BK
★CZ-6EB1
標準価格88,000円(税別)

スピーカー

アンプ内蔵
スピーカーシステム(2本1組)
AN-S100
標準価格36,600円(税別)

システムラック

システムラック
(CZ-600C/601C/602C/603C/604C/611C/612C/613C/623C/634C/644C用)
CZ-6SD1
標準価格44,800円(税別)

■本広告に掲載しております拡張ボード類のうち、CZ-634C/644Cの16MHzモードで動作しないものが一部あります。 ★印の商品は在庫僅少です。
■製品改良のため仕様の一部を予告なく変更することがあります。またこの広告の色調は印刷のため実物とは多少異なる場合もありますのであらかじめご了承ください。

CZ-600C用)、CZ-6BE1B 標準価格28,000円(税別・CZ-601C、CZ-611C、652C、653C、662C、663C用)を増設してください。 ※7 CZ-600C、601C、602C、603C、611C、612C、613Cに装着の場合、I/Oスロット2に装着ください。CZ-652C、653C、662C、663Cに装着の場合はI/Oスロット4に装着ください。また、CZ-6BG1、6BU1、6BL1、6BL2、6BN1などのボードは、接続コネクタとの関係で本ボードとの併用はできませんのでご注意ください。なお、本ボードはX68000用OS Human 68K ver.2.0以上にてご使用ください。 ※8 モデムユニットCZ-8TM2に同梱のソフトはX1/X1ターボシリーズ用です。

子機器事業本部液晶映像システム事業部第2商品企画部 〒162 東京都新宿区市谷八幡町8番地 ☎(03)3260-1161(大代表)

SHARP

X68000 XVIデビュー記念キャンペーン

「夢、創ります。山下章氏プロデュース」

第1回全日本X68000 芸術祭開催

作品募集中!

X68000アイドル山下章氏司会、進行による
ユーザー参加型作品コンテスト

■主催：シャープ株式会社 電子機器事業本部 システム機器営業部
■共催：シャープエレクトロニクス販売株式会社各統轄営業部
東京中央シャープ販売㈱・浪速シャープ電機㈱・沖縄シャープ電機㈱
■協賛：出版社・ソフトハウス・サードパーティ・主要販売店

X68熱のヒートアップは、当日会場で!

7月21日の四国地区大会をトップに、X68000芸術祭がヒートする。
夏の暑さをふっとばす過熱ぶりを、会場でじかにご体験ください。ご来場をお待ちしています。

四国地区大会いよいよ開催 (7/21㊐於・高松)

北海道地区大会締切り迫る! 7/26㊎まで (当日はぜひ会場まで8/4㊐於・札幌)

東北地区大会締切りも間近 8/23㊎まで

〈作品応募要項〉

■**作品基準**：パーソナルコンピュータ(メーカー、機種を問わず)で制作した、オリジナル未発表のプログラム、グラフィックス、コンピュータ・ミュージック等であること。なお応募作品(制作に使用したアプリケーション・ソフト等以外の部分)の著作権は、すべてシャープ㈱に帰属します。■**部門**：①**ゲーム部門**、②**ミュージック部門**(自作の曲/一般曲・ゲームミュージックのアレンジ等、MIDI使用も可。)③**グラフィックス部門**(Z'sSTAFF PRO-68K、DOGA等のツールを使用して描いたものなど画面上に表示されるグラフィックスなら何でも可。)④**その他部門**：ユーティリティ/一発ギャグ/パフォーマンス/ビジネス利用/その他)※応募は、1部門につき1人1作。1人複数部門応募は可。また団体制作も可。■**応募資格**：各予選ブロックの地域の住人であること。■**応募方法**：フロッピー・ディスクに住所/氏名/年齢/職業(学校名・学年)/電話番号/開発に要した期間/開発に使用・利用したツール名/セールスポイント/取り扱い上の注意/動作に必要とする特殊機材を添え、各地区の応募先まで郵送してください。締め切りはその地区の地区大会開催日の2週間前(必着)です。■**賞・賞品**：〈地区予選〉●各地区大会大賞(1点)トロフィー、賞状、副賞●入選(首都圏3点、近畿2点、中部・九州1点、他地区なし)賞状、副賞●協賛各社賞・賞状、副賞〈全国大会〉●第1回全日本X68000芸術祭グランプリ(1点)トロフィー、賞状、副賞：光磁気ディスクユニット(CZ-6MO1)ペアでの海外旅行(旅行クーポン)●ゲーム・ミュージック・グラフィック等各部門賞・賞状、副賞●協賛各社賞・賞状、副賞

※詳細は店頭のチラシをご覧ください。

	開催地	開催日	会場	入選枠	対象都道府県	応募・問い合わせ先	締切日
7月	四国(高松)	7月21日㈰	シャープ高松ビル 5Fイベントホール 高松市朝日町6-2-8 ☎0878-23-4860	大賞1点	徳島・香川・愛媛・高知	〒760 高松市朝日町6-2-8 シャープエレクトロニクス販売㈱四国統轄(営) パソコン担当、辻井部長・細川係長 ☎0878-23-4860㈹	7月5日㈮
8月	北海道(札幌)	8月4日㈰	シャープ札幌ビル 4Fホール 札幌市西区二十四軒1条7-3-17 ☎011-642-8111	大賞1点	北海道	〒063 札幌市西区二十四軒1条7-3-17 シャープエレクトロニクス販売㈱ 北海道統轄(営) パソコン担当、長谷田担当 ☎011-642-8111㈹	7月26日㈮
9月	東北(仙台)	9月8日㈰	シャープ仙台ビル 4Fホール 仙台市若林区卸町東3-1-27 ☎022-288-9111	大賞1点	青森・山形・岩手・福島・宮城・秋田	〒983 仙台市若林区卸町東3-1-27 シャープエレクトロニクス販売㈱ 東北統轄(営) パソコン担当、岡本部長・阿部課長 ☎022-288-9111㈹	8月23日㈮
9月	中国(広島)	9月15日㈰	広島市西区民センター 3F大会議室A 広島市西区横川新町6-1 ☎082-234-1960	大賞1点	鳥取・島根・岡山・広島・山口	〒731-01 広島市安佐南区西原2-13-4 シャープエレクトロニクス販売㈱ 中国統轄(営) パソコン担当、青木部長・石井担当 ☎082-874-2282㈹	8月30日㈮
9月	北関東(宇都宮)	9月22日㈰	護国会館 平安の間 宇都宮市陽西町1-37 ☎0286-22-3180	大賞1点	茨城・群馬・栃木	〒320 宇都宮市不動前4-2-41 シャープエレクトロニクス販売㈱ 北関東統轄(営) パソコン担当、岩田部長・川俣係長 ☎0286-35-1151㈹	9月6日㈮
10月	神奈川(横浜)	10月6日㈰	神奈川県労働総合センター 5F大講堂 横浜市磯子区中原1-1-28 ☎045-773-2250	大賞1点	神奈川	〒235 横浜市磯子区中原1-2-23 シャープエレクトロニクス販売㈱ 神奈川統轄(営) パソコン担当、常次部長 ☎045-753-5501㈹	9月20日㈮
10月	中部(名古屋)	10月20日㈰	シャープ名古屋ビル 7Fホール 名古屋市中川区山王3-5-5 ☎052-323-5111	大賞1点 入選1点	静岡・愛知・長野・岐阜・三重	〒454 名古屋市中川区山王3-5-5 シャープエレクトロニクス販売㈱ 中部統轄(営) パソコン担当、山口課長 ☎052-323-5111㈹	10月4日㈮
11月	北陸(金沢)	11月3日㈰	労済会館 金沢市西念1-12-22 ☎0762-23-5911	大賞1点	富山・石川・福井	〒921 石川県石川郡野々市町字御経塚町1096-1 シャープエレクトロニクス販売㈱北陸統轄(営) パソコン担当、小林担当 ☎0762-49-1181㈹	10月18日㈮
11月	近畿(大阪)	11月10日㈰	シャープ本社 4F第一集会室 大阪市阿倍野区長池町22-22 ☎06-621-1221	大賞1点 入選2点	滋賀・京都・大阪・兵庫・奈良・和歌山	〒556 大阪市浪速区恵美須西1-2-9 シャープエレクトロニクス販売㈱ 近畿統轄(営) パソコン担当、岡本課長・細川係長 ☎06-631-1181㈹	10月25日㈮
11月	首都圏(東京)	11月24日㈰	シャープ東京支社 8Fエルムホール 東京都新宿区市ヶ谷八幡町8 ☎03-3260-1161	大賞1点 入選3点	埼玉・山梨・千葉・新潟・東京	〒162 東京都新宿区市ヶ谷八幡町8 シャープエレクトロニクス販売㈱ 首都圏統轄(営) パソコン営業部、福井部長・前田課長 ☎03-3266-8248	11月8日㈮
12月	九州(福岡)	12月14日㈯	KC会館 2F大ホール 福岡市博多区博多駅前3-4-2 ☎092-451-5971	大賞1点 入選1点	福岡・佐賀・長崎・熊本・大分・宮崎・鹿児島・沖縄	〒816 福岡市博多区井相田2-12-1 シャープエレクトロニクス販売㈱ 九州統轄(営) パソコン営業部、北山部長・岩崎課長 ☎092-501-6806	11月29日㈮
2月	補選(大阪)	2月9日㈰(予定)	(未定)	入選2点	全国	〒545 大阪市阿倍野区長池町22-22 シャープ株式会社 電子機器事業本部 システム機器営業部 ☎06-621-1221㈹	1月24日㈮(予定)

協賛雑誌(50音順)：I/O(工学社)・アスキー(アスキー)・Oh!X(ソフトバンク)・コンプティーク(角川書店)・POPCOM(小学館)・マイコン(電波新聞社)・マイコンBASICマガジン(電波新聞社)・LOGIN(アスキー)
協賛メーカー：ローランド株式会社

68買ったらEXEクラブに入ろう!

本体同梱の入会申込ハガキを送るだけで、無料入会。3つのメリット!
メリット1：会員No.入りオリジナル**会員証電卓**がもらえる。
メリット2：各種フェアご優待・イベントご案内等、数々の特典あり。
メリット3：X68000の活用情報が手に入る「**EXEおみこし活動**」に参加できる。
※「申込ハガキををなくしてしまった」という方は、右記「おみこし活動隊」までお電話ください。

EXEおみこし活動とは?
コミュニケーションペーパー「おみこしPRESS」を通じて会員同士が情報を交換、どこまでもX68000を使いこなして盛り上がってしまおう!というのが、その目的。68へのラブコール、会員独自のテクニック・活用法など、あなたの68自慢を「おみこし活動隊」までどうぞ。会員メッセージは随時「おみこしPRESS」に掲載します。

さらに熱心な会員のために、「**おみこしかつぎ人**」制度も設けました。「かつぎ人」3つのメリットは…❶X68000情報交換会「**おみこしかつぎ人の集い**」に参加できる。❷68最新ソフト・各周辺機器が一覧できる「**ソフトウェア・フィールド**」を半年1回送付。❸「**おみこしPRESS**」毎号送付。「かつぎ人」になれば68ユーザーとして一層充実すること間違いなしです。

●「おみこしかつぎ人」になるには、年会費(おみこしかつぎ代)が必要です。個人入会3,000円/グループ入会(5人1組)2,500円・郵便振込にて申込受付。●詳細は店頭の「おみこしPRESS」をご覧になるか、または「おみこし活動隊」にお電話ください。
おみこし活動隊☎(06)886-0354

●お問い合わせは…シャープ㈱電子機器事業本部システム機器営業部 〒545 大阪市阿倍野区長池町22番22号 ☎(06)621-1221(大代表) シャープ株式会社

TX-80 ¥108,000

- 80MBタイプハードディスクユニット
- 平均アクセスタイム20ms
- SASI、SCSI両対応(SCSIモードで使用の場合はシャープ製SCSIインターフェースボードCZ-6BS1が必要)
- 本体塗装色はブラックとグレーの2色を用意
- 60(W)×120(H)×295(D)㎜の省スペース設計

TX-130 ¥138,000

- 130MBタイプハードディスクユニット
- 平均アクセスタイム20ms
- 完全SCSI対応(SUPERシリーズ及びXVIシリーズ以外で使用の場合はシャープ製SCSIインターフェースボードCZ-6BSIが必要)
- 本体塗装色はブラックとグレーの2色を用意
- 60(W)×120(H)×295(D)㎜の省スペース設計

TX-180 ¥185,000

- 180MBタイプハードディスクユニット
- 平均アクセスタイム20ms
- 完全SCSI対応(SUPERシリーズ及びXVIシリーズ以外で使用の場合はシャープ製SCSIインターフェースボードCZ-6BSIが必要)
- 本体塗装色はブラックとグレーの2色を用意
- 60(W)×120(H)×295(D)㎜の省スペース設計

アイテック株式会社

本社 〒550 大阪市西区新町1丁目25番14号 TEL:(06)532-0216 FAX:(06)532-7253
関西営業所 〒550 大阪市西区新町1丁目25番14号 TEL:(06)532-0120 FAX:(06)532-0543
関東営業所 〒275 千葉県習志野市谷津1丁目12番5号 TEL:(0474)77-7564 FAX:(0474)73-2759
名古屋営業所 〒460 名古屋市中区大須2丁目28番31号 TEL:(052)212-1487 FAX:(052)212-1627

インフォメーションセンター TEL:(06)532-0320

X68000

8月24日発売!!

冒険

ファンタジーロールプレイングゲーム

ロードス島戦記©

灰色の魔女

原作/安田　均・水野　良

オリジナルキャラクターデザイン/出渕　裕　標準価格**9,800**円

Record of Lodoss War

【ユーザーズテレホン ☎大阪06(315)8255
平日の午後1時半から6時の間は、お問い合せに直接お答えします。その他の時間と土・日・祝日はまるまる24時間録音できるテープサービスです。】

◆標準価格に消費税は含まれておりません。お買上げの際に別途消費税をお支払い下さい。

◆通信販売ご希望の方は、住所・氏名・電話番号・商品名・機種名・メディアを明記の上、現金書留または郵便振替(大阪8-303340)にてお申し込み下さい。送料は無料ですが、標準価格に消費税の3%を加えた金額をお送り下さい。

株式会社エム・エー・シー　ハミングバードソフト

〒530 大阪市北区曽根崎2丁目2番15号

Mu-1 Musicstudio Super

[ミューワン スーパー]

定価 ¥39,800(税別)

“Musicstudio”スペシャルバージョン新登場!

Mu-1 Superは、マルチレコーダー“Musicstudio PRO-68K Ver2.0”にMu-1のミュージ郎コンバート機能、グラフィック機能を合体させ、さらにパワーアップしたスペシャルバージョンです。

◆ 特 長 ◆

❶ミュージ郎(98用5"2HD)データコンバートはもとより、マックやアタリなど世界のMIDIソフトと双方向のデータコンバートが可能な“スタンダードMIDIファイル”対応 ❷Sound Canvas(ローランド社新音源SC-55)のGS規格に対応した新デザイン“MTRウィンドウ” ❸CM-32L、64、MT-32の音色を自由に作り変えられる“LA音源音色エディタ搭載 ❹ドラム音色変更可能な52チャンネルミキサー搭載(CM-32L、64、MT-32専用) ❺他の音源と差し替えに便利なCM&MT音源専用ミュート機能 ❻LA音源用音色ライブラリ収録(CM-32L、64、MT-32専用) ❼内蔵FM音源、内蔵ADPCM音源対応 ❽リアルタイム録音機能 ❾ステップ入力、エディット強化(UNDO機能等) ❿国本佳宏氏のDEMO曲とKUNTA氏のグラフィックデータ収録

●本ソフト動作には、メインメモリ2MB及びMIDIボードが必要です。●Mu-1を既にお持ちのお客様には有償バージョンアップサービスを実施します。

新デザイン“MTRウィンドウ”

52chドラムミキサー

LA音源音色エディタ

スピーディーなステップ入力

Mu-1 Magazine

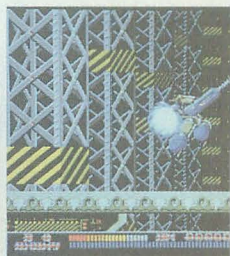
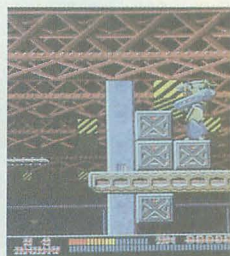
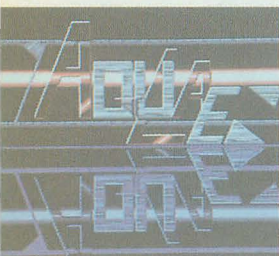
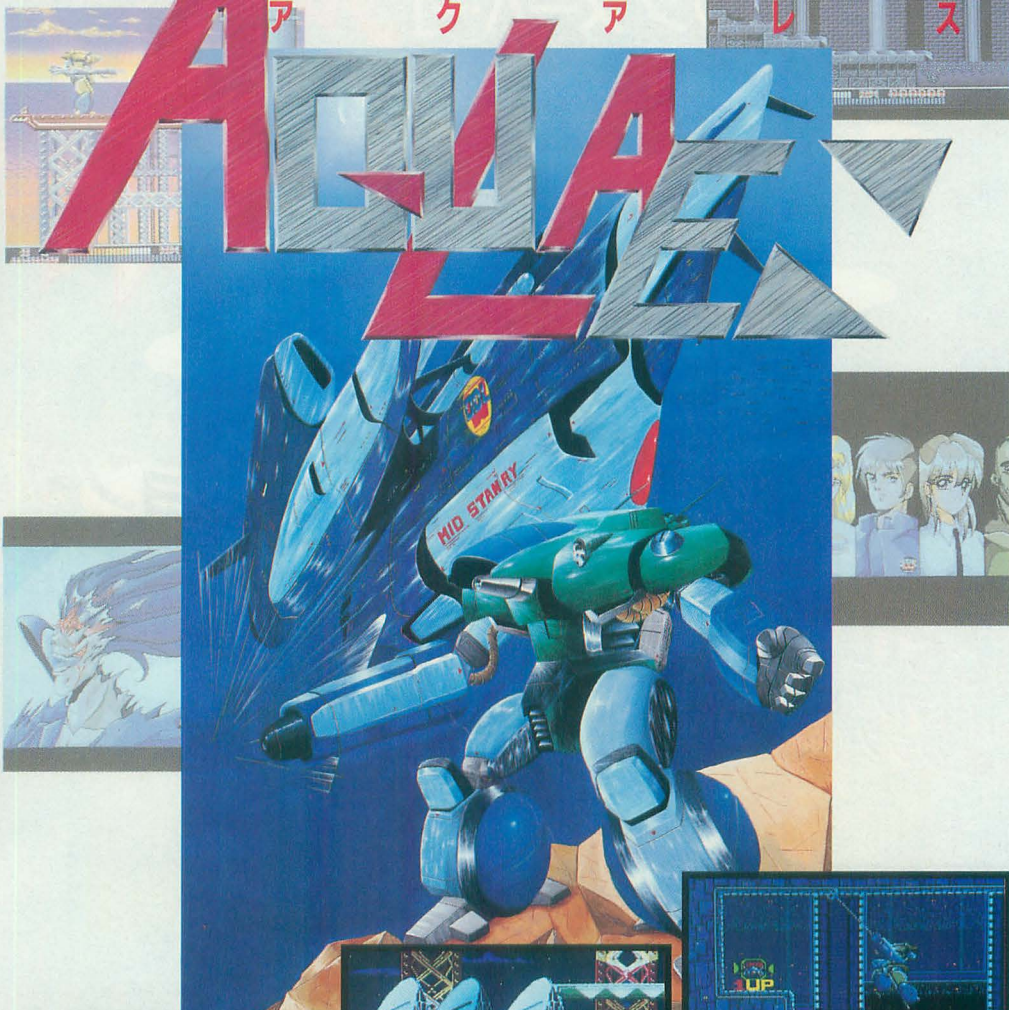
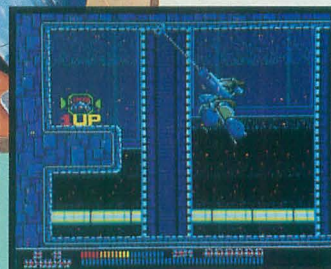
ミューワンマガジン
Vol.1
8月下旬発売予定 ¥9,800(税別)

1. ソングファイルクラシック
   特集 モーツァルト(CM-64、MT-32+LA音源対応)
   フィガロの結婚 序曲/魔笛 序曲/ドン・ジョバンニ序曲/ホルン協奏曲 第4番(1楽章 アレグロ・モデラート 2楽章 ロマンツェ・アンダンテ 3楽章 ロンド・アレグロ・ヴィバーチェ)
2. 鳥山敬治LA音源音色ライブラリVol.1
3. KUNTAさんのイラストデータ
4. スタンダードMIDIファイル読み込みコンバート
5. Mu-1 Super体験版
   "Mu-1 聴くだけ"搭載●Mu-1のSNGファイルの再生、チェインプレーが可能●LA音源の音色変更可能
6. その他おまけ機能

＊本ソフト動作には、MIDIボードか必要です

〒154 東京都世田谷区池尻3-21-28 池尻成和ビル 03(3419)8839

●Musicstudio、PRO-68K、Ver2.0はシャープ㈱の登録商標です。
●ミュージ郎はローランド㈱の登録商標です。

X68000
オリジナル・ロボットアクション
深海3万8千メートル
静寂なる暗闇に戦慄のシンフォニーが響きわたる
アクアレス
AQUALES
SHIELD
1UP
★狂える瀑布を乗り越えろ。
★テロによる衛星強奪を阻止せよ。
★要塞の奥に待ちうけるものは?
全8面
24ステージ
※写真は開発中のものです。
お求めは、全国パソコン専門店で。
取扱店がない場合は、住所・氏名・電話番号と商品名を明記し、価格に消費税を加算の上、当社宛に現金書留にてお申し込み下さい。
〈発売元〉
エグザクト
EXACT
新潟市米山3-2-11 MKD.5ビル
☎(025)247-9160(代)
8月発売予定
8,700円
〈税別〉

NEW 3D GOLF SIMULATION

Masters®

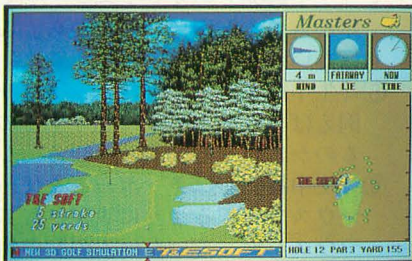
LICENSED BY
AUGUSTA NATIONAL GOLF CLUB

遥かなるオーガスタ™

はるかなるオーガスタ

オーガスタ・ナショナル・ゴルフ・クラブと正式契約

X68000版
好評発売中!!

(5"2HD 3枚組)
要2M RAM
■標準価格 ¥12,800(税別)

- ●実際にゴルフコースに立った状態と同じ視野でプレイ可能
- ●どの地点にきても全方向の視野画面をリアルタイム3D表示
- ●ホールすべてにアンジュレーション（起伏）を３Ｄで表示
- ●ボールの落下地点の状態によってバウンド、転がり等が本物同様に変化
- ●ストロボアクションモードでボールの軌跡を確認可能
- ●バックスピン・トップスピンも可能
- ●プレイモードは３種類。ストロークプレイ、マッチプレイ、トーナメントプレイ
- ●キャディーは４人の中から選択
- ●初心者でも気軽に楽しめるスローモード機能あり
- ●スコア・各種個人データ等を自動保存、プリントアウトも可能
- ●マウスのみですべて簡単操作
- ●３１ＫＨｚ／１５ＫＨｚ両モード対応、ビデオ出力で大画面プレイ可能
- ●ＡＤＰＣＭによるリアルなサウンド
- ●その他機能満載

コースデータVol.2

NEW 3D GOLF SIMULATION

EIGHT LAKES G.C.™

T&E SOFT ORIGINAL COURSE

エイトレイクスG.C.は、その設計に充分な時間をかけ、細部にわたって練り上げられた戦略重視のオリジナルコースです。

- ●８つの湖が効果的にレイアウトされた美しい18ホール
- ●湖、森林、谷、丘陵、そして砂浜等、変化に富んだ難コース
- ●目にも鮮やかな桜並木が水面に影を落す
- ●各ホールとも多彩な攻め方が要求される、戦略重視のテクニカルコース
- ●速いグリーンがより高度なパッティングテクニックを要求する
- ●国際色豊かな４人の女性キャディーが登場

X68000版
好評発売中!

(5"2HD 2枚組) ■標準価格 ¥5,800(税別)

※「EIGHT LAKES G.C.」は、「遥かなるオーガスタ」が必要です。

スリーアイランズ"と呼ばれる名物ショートホール

このマークはT&E SOFTの商標です
POLYSYS搭載の3Dソフトには、このマークが表示されます

Technology & Entertainment Software

株式会社 ティーアンドイーソフト

〒465 名古屋市名東区豊が丘1810番地 PHONE:052-773-7770

●3Dゴルフに関するお問い合わせは、NEW 3D GOLF事務局まで PHONE:052-773-7757

それでも君は、

生中継68 に

「生中継68」の前評判が高いらしいが、オレはだまされないぞ。
というのは、以下の9つの事実を知っているからだ。

①「生中継68」のゲーム画面は攻撃側と守備側に分割される。
画期的な画面構成だというが、これでは他の野球ゲームで養ったカンが通用しないのではないか。不安だ。

②「生中継68」のエディットモードは、設定する項目が細かすぎる。
エディットしなくてもちゃんとゲームは楽しめるのだから、ここまで緻密な設定機能がはたして必要なのかどうか。

③「生中継68」では、チームのユニフォームの色やデザインを選べるのは当然としても、それにあわせてチームマークのカラーまでコーディネイトされてしまうのは大きなお世話ではないか。

④「生中継68」のオープニングデモは確かに迫力はある。が、ゲーム自体がきちんと面白いのだから、こんなところまで凝るのはヤリ過ぎではないか。

⑤「生中継68」のグラフィックはリアルすぎる。
例えばデッドボールはどうするのだ。こんなリアルな映像で再現されるのかと思うと、おお、観る前から痛さに寒気がするではないか。

⑥「生中継68」はキーボードを使わなくても、全てのエディット操作をマウスでできるように設定されている。
マウスを使っていない人の胸の痛みを考えたことがあるのか。

⑦「生中継68」には「操作方法ミニPOP」がついているそうだ。
そんなオマケをサービスするというのは、企業努力を見せびらかしたいシタゴコロが見え見え。かた腹痛いわ。

⑧「生中継68」のゲーム性は「68版野球ゲームの決定版」を狙った完成度だそうだが、それは商売上の理由よりも開発担当者の個人的なこだわりでやっているとしか思えない。

⑨この広告だってヘンだ。

※操作方法ミニPOP…パッケージの中に入っているオマケ。
謙遜ではなく、本当に大したモノではない
この場合のPOPは、Point Of Play の略

コナミ株式会社 東京本社／〒101 東京都千代田区神田神保町3丁目25　大阪支店／〒561 大阪府豊中市庄内栄町4丁目23-18
札幌営業所／〒060 札幌市中央区北一条西5丁目2-9　福岡営業所／〒810 福岡市中央区天神2丁目8-30

期待するのか？

それでも期待してくれる人は、開発者にはげましのお便りを出そう！
X68000
生中継68™
7月30日発売
※発売日が変わりました。

全国通販

SHARP 認定 PPO-SHOP **O.A.ランド**

(TEL) 03-3770-8855

■アフターサービス万全のサポート体制
●下取・買取は電話で見積りしております。責任を持って下取りさせて頂きます。

営業時間
平日………AM10:00〜PM7:00
土日・祭日…AM10:00〜PM6:00

▶7・15〜8・14

SHARPのことなら なんでもおまかせ!!

大徳買セール! 安く値切ってネ。(本体セット:送料・消費税込み)
お電話下さい。㊙価格をお知らせいたします。

流通事情により、広告表示価格は、お安くなる場合がありますので、ドンドンお電話下さい。

CYBER STICK
■CZ-8NJ2 (定価¥23,800)
OAランド特価 ▶¥18,000

電子手帳 ●見やすい漢字4桁表示!! 情報化時代の必需品!!
■PA-9500(¥48,000)…▶特価¥38,000
■PA-8500(¥28,000)…▶特価¥15,000
■PA-7500(¥22,000)…▶特価¥12,000

SHARP X68000シリーズセット(送料・消費税込み)

X68000XVI

①CZ-634-TN+CZ-614D-TN
定価合計¥503,000

回数	月額
12回	¥33,100
24回	¥17,600
36回	¥12,200
48回	¥9,600

②CZ-634C-TN+CZ-607D-TN
定価合計¥467,800

回数	月額
12回	¥30,800
24回	¥16,300
36回	¥11,400
48回	¥8,900

③CZ-634C-TN+CZ-606D-TN
定価合計¥447,800

回数	月額
12回	¥29,500
24回	¥15,700
36回	¥10,900
48回	¥8,500

X68000XVI-HD

①CZ-644C-TN+CZ-614D-TN
定価合計¥653,000

回数	月額
12回	¥42,800
24回	¥22,700
36回	¥15,800
48回	¥12,400

②CZ-644C-TN+CZ-607D-TN
定価合計¥618,700

回数	月額
12回	¥40,600
24回	¥21,500
36回	¥15,000
48回	¥11,700

③CZ-644C-TN+CZ-606D-TN
定価合計¥597,800

回数	月額
12回	¥39,300
24回	¥20,800
36回	¥14,500
48回	¥11,400

■CZ-634C■ 特価 ¥TEL下さい!!

■CZ-644C■ 特価 ¥TEL下さい!!

XVIお買い上げの方に❶ニュージーランドストーリー ❷V-BALL ❸ジョイカード(連射式) ❹ディスケット20枚プレゼントいたします!!

現金でお買い上げの方には、さらに超特値でお出ししてます。ぜひ一度TEL下さい!!

X68000SUPER/SUPER-HDシリーズ

★CZ-604C+CZ-606D…特価¥290,000
★CZ-623+CZ-606D……特価¥375,000

X68000メーカー展示品スペシャルセット 限定

★CZ-634C+CZ-606D…特価¥323,000
★CZ-644C+CZ-606D…特価¥435,000
★CZ-8PC5……………特価¥69,000

上記組合せのディスプレイ(モニター)変更自由!! 詳しくは、お電話にてお問い合せ下さい!!

■期間中、セットでお買い上げの方には、①ジョイカード(連射式)と②テトリスやドルアーガの塔などの入ったゲームパックをプレゼント!!

周辺機器コーナー 電話で値切ろう。

プリンターセットコーナー

①CZ-8PC5 NEW 定価¥96,800
●48ドット●熱転写カラー 漢字プリンター
大特価TEL下さい!!

②CZ-8PK10(24ピン漢字プリンター136桁)
定価¥97,800…特価¥71,000

③CZ-8PG1(24ピンカラー漢字プリンター80桁)
定価¥130,000…特価¥93,000

④CZ-8PG2(24ピンカラー漢字プリンター136桁)
定価¥160,000…特価¥114,000

OAランド特選品!!

■IO-735X(定価¥248,000)
●カラーイメージジェットプリンター
特価¥165,000

X68000用ハードディスク

■SCSIタイプ
●アイテック
①TX-80S (¥108,000)…特価¥78,500
②TX-130S(¥138,000)…特価¥98,500
③TX-180S(¥185,000)…特価¥132,000

■SASIタイプ
●アイテック
①ITX-680(¥198,000)…特価¥68,000
●ロジテック
①SHD-40(¥99,800)…特価¥60,000

※X68000SUPER/XVI以外の機種では、SCSIボードが必要となります。

★SCSIボード………特価¥22,000
★光ディスク…………特価¥320,000

X68000用周辺機器コーナー

①CZ-6VT1(カラーイメージユニット)
定価¥69,800…特価¥51,500
②CZ-8NS1(カラーイメージスキャナー)
定価¥188,000…特価¥135,000
③CZ-6BM1(MIDIボード)
定価¥26,800…特価¥20,000
④CZ-6BE2A(2MB増設RAMボード)
定価¥59,800…特価¥44,000
⑤CZ-6BE2B(2MB増設RAM)
定価¥54,800…特価¥40,500
⑥CZ-6BP2(数値演算プロセッサ)
定価¥45,800…特価¥33,800
⑦CZ-6EB1(拡張I/Oボックス=4スロット)
定価¥88,000…特価¥65,000
⑧CZ-6BP1(数値演算プロセッサボード)
定価¥79,800…特価¥59,000

《計測技研》増設メモリ&プロセッサ

●高速増設メモリと数値演算プロセッサが一つのボードになった!!●

● KGB-X68PRKII-02(¥55,000)……特価¥42,800
● PRKII-04(¥90,000)……特価¥70,200
● PRKII-06(¥125,000)……特価¥97,500
● PRKII-08(¥160,000)……特価¥124,800
● PRKII-12(¥85,000)……特価¥66,300
● KGB-X68PRKII-14(¥120,000)……特価¥93,600
● PRKII-16(¥155,000)……特価¥121,000
● PRKII-18(¥190,000)……特価¥148,000
● MC-6888 IRC(¥38,000)……………特価¥28,500

I・Oデータ増設RAMボード

■PIO-6BE1-A (1MB) 定価¥25,000 特価¥17,300
■PIO-6BE2-2M (2MB) 定価¥50,000 特価¥33,500
■PIO-6BE4-4M (4MB) 定価¥88,000 特価¥58,500

■OAランド推奨ソフト

Ⓐ Easy Paint SX 68K (CZ-263GW) 特価TEL下さい!!
Ⓑ Music studio PRO 68K (CZ-261MS) 定価¥28,800 特価TEL下さい!!
Ⓒ Print Shop V.2 (CZ-265HS) 特価TEL下さい!!
Ⓓ Multiword PRO 68K (CZ-225BS) 定価¥32,000 特価¥24,000
Ⓔ CZ-245LS (C-コンパイラII) 定価¥44,800 特価¥33,500
Ⓕ Telepotion PRO 68K (CZ-258BS) 定価¥22,800 特価¥18,000

通信販売のご案内

全国通販

■銀行振込で申し込みの方は商品名及びお客様の住所・氏名・電話番号をお知らせ下さい。
〔振込先〕第一勧業銀行 渋谷支店
普通No.1163457 ㈱オーエーランド

■年中無休です!!

■現金書留で送金されるお客様は電話番号と商品名、数量を明記して同封して下さい。■クレジットでご購入を希望される方は申し込み用紙をお送り致しますのでご記入の上返送して下さい。20才以上の方は、原則として保証人不要です。クレジットは1〜60回払で月々5,000円よりご自由に設定できます。

クレジット表

3回	3.5%	6回	4.5%	10回	6%	12回	6%	15回	8.5%	18回	11%	20回	12%
24回	12.5%	30回	17%	36回	17.5%	42回	22.5%	48回	23%	54回	29%	60回	29.5%

㈱オー・エー・ランド

〒150 東京都渋谷区桜丘町3-13 アルカディア2F
☎(03)3770-8855

関東エリアの送料は、1個につき¥1,000です。 FAX(03)3770-7080

★全商品保証書付。専門のアドバイザーが、お客様のニーズに対応します。
★初期不良・輸送トラブル等に迅速に対応し、即交換させていただきます。

低金利クレジットをご利用下さい。平日AM10時〜PM7時、土日・祭日AM10時〜PM6時迄ガンバッテま〜す!!

■ゲームソフト、ビジネスソフト2割、3割引きは、当り前で〜す。お問い合せ下さい。

■表示価格は、税別表示です。詳しくは、お電話にて、お問い合せ下さい。掲載の価格は、6月下旬現在です。

■特に人気のある商品によっては、しばらくお待ち願うことがありますのでご了承下さい!!

X68000 ラストチャンス!!

■SUPER／PROII／SUPER-HD

◀大人気!!*大戦略II シュミレーションゲーム
プレゼント
★JOY CARD (連射式)×2個
★MD-2HD 10枚
(定価￥9,800)

■超低金利クレジットをご利用下さい。1回～60回払い、頭金ナシ/ボーナス一回及び2回払いOKです。

■ビッグバーゲンセール実施中!!ゲームソフト(ビジネス)新製品続々入荷中!!

限定

■SUPER(定価￥348,000)
CZ-604C-TN

■PRO II(定価￥285,000)
CZ-653C-BK/GY

■SUPER-HD(定価￥498,000)
CZ-623C-TN

15型カラーディスプレイTV

CZ-614D-TN
定価￥135,000

14型カラーディスプレー

CZ-606D(GY/BK/TN)
定価￥79,800

21型カラーディスプレイ

CU-21HD
定価￥148,000

CZ-8NJ2 限定
●インテリジェントコントローラ
定価￥23,800
超特価￥18,000

(送料・消費税込)

Ⓐ CZ-604C＋CZ-614D…定価合計￥483,000▶ **￥329,000**

12回	￥29,000	24回	￥15,400	36回	￥10,700	48回	￥ 8,400	60回	￥ 7,200

Ⓑ CZ-653C＋CZ-614D…定価合計￥420,000▶ **￥278,000**

12回	￥24,500	24回	￥13,000	36回	￥ 9,000	48回	￥ 7,100	60回	￥ 6,100

Ⓒ CZ-623C＋CZ-614D…定価合計￥633,000▶ **￥408,000**

12回	￥36,000	24回	￥19,100	36回	￥13,300	48回	￥10,400	60回	￥ 9,000

Ⓓ CZ-604C＋CZ-606D…定価合計￥427,800▶ **￥290,000**

12回	￥25,600	24回	￥13,500	36回	￥ 9,400	48回	￥ 7,400	60回	￥ 6,400

Ⓔ CZ-653C＋CZ-606D…定価合計￥364,800▶ **￥238,000**

12回	￥21,000	24回	￥11,100	36回	￥ 7,700	48回	￥ 6,000	60回	￥ 5,200

Ⓕ CZ-623C＋CZ-606D…定価合計￥577,800▶ **￥375,000**

12回	￥33,100	24回	￥17,500	36回	￥12,200	48回	￥ 9,600	60回	￥ 8,300

Ⓖ CZ-604C＋CU-21HD‥定価合計￥496,000▶ **￥340,000**

12回	￥30,000	24回	￥15,900	36回	￥11,000	48回	￥ 8,300	60回	￥ 7,500

Ⓗ CZ-653C＋CU-21HD‥定価合計￥433,000▶ **￥285,000**

12回	￥25,100	24回	￥13,300	36回	￥ 9,300	48回	￥ 7,300	60回	￥ 6,300

Ⓘ CZ-623C＋CU-21HD‥定価合計￥646,000▶ **￥425,000**

12回	￥37,500	24回	￥19,900	36回	￥13,800	48回	￥10,800	60回	￥ 9,400

★本体セットは、1ヶ月間だけの大特価セール!!
★クレジット価格は、消費税込みですョ。ご利用下さい!!

X68000ソフト大セール実施中!!(ゲームソフト25～30%OFF)　送料￥500

〈グラフィック〉●Z's STAFF PRO68K Ver.2.0 (シャフト)定価￥58,000 ……特価￥38,500

〈開発ツール〉●C-コンパラPRO68KV.2 定価￥44,800 CZ-245IS ……特価￥33,000

〈データベース〉●CARD PRO68K Ver.2.0 定価￥29,800 CZ-253BS ……特価￥21,000

〈グラフィック〉●C-TRACE＋ 定価￥198,000 ……特価￥145,000

〈C言語〉●C & Professional Pack 定価￥58,000 ……特価￥40,500

〈音楽〉●Music studio PRO68K Ver.2.0 定価￥28,800 CZ-261MS ……特価￥21,300

〈CGツール〉●CANVAS PRO68K 定価￥29,800 CZ-249GS ……特価￥22,200

〈ワープロ〉●Multiword PRO68K 定価￥32,000 CZ-225BS ……特価￥23,800

〈通信〉●Tlepotion PRO68K 定価￥22,800 CZ-258BS ……特価￥17,000

型名	商品	定価	特価	型名	商品	定価	特価
CZ-212BS	BUSINESS PRO-68K	￥ 68,000	￥ 48,000	EW		￥ 38,000	￥ 29,800
CZ-213MS	MUSIC PRO68K	￥ 18,800	￥ 13,500	G-68K		￥ 14,800	￥ 11,400
CZ-214MS	SOUND PRO-68K	￥ 15,800	￥ 11,500	E-68		￥ 19,800	￥ 15,300
CZ-215MS	Sampling PRO-68K	￥ 17,800	￥ 12,800	CZ-255GS	CANVASドローグラフィックLIB	￥ 8,800	￥ 29,600
CZ-219SS	OS-9/X68000	￥ 29,800	￥ 21,000	CZ-256GS	CANVASドローグラフィックVOL.2	￥ 8,800	￥ 6,600
CZ-220BS	DATA PRO-68K	￥ 58,000	￥ 41,000	CZ-260LS	XBAS to CHECKER PRO68K	￥ 9,800	￥ 6,600
CZ-223CS	Communication PRO-68K	￥ 19,800	￥ 14,300	CZ-259SS	SX-WINDOW Ver.1.0	￥ 6,800	￥ 7,500
CZ-224LS	THE 福袋 V2.0	￥ 9,900	￥ 7,500	CZ-234LS	AI-68K	￥188,000	￥ 5,000
CZ-241BS	システム手帳リフィル集	￥ 9,800	￥ 7,500	CZ-251BS	ハイパーワード	￥ 39,800	￥139,000
CZ-242BS	活用フォーム集	￥ 9,800	￥ 7,500		KAMIKAZE	￥ 68,000	￥ 29,800
CZ-244SS	Homan 68K Ver2.0	￥ 9,800	￥ 7,500		デジタルクラフト	￥ 39,800	￥ 45,400
CZ-247MS	MUSIC PRO-68K〔MIDI〕	￥ 28,800	￥ 20,800		サイクロエキスプレスα68	￥ 97,000	￥ 28,000
CZ-240BS	Stationery PRO-68K	￥ 14,800	￥ 11,500		C-TRACE 68 Ver3.0	￥ 78,000	￥ 73,000
CZ-243BS	CYBER NOTE PRO-68K	￥ 19,800	￥ 15,200		C-TRACE TP＋	￥398,000	￥ 69,500

熱転写カラー漢字プリンター (送料￥1,000)

■CZ-8PC5

●48ドット
●熱転写カラー漢字プリンター
定価￥96,800
特価￥TEL下さい!!(ケーブル用紙付)

ハードディスク (送料￥1,000)

■アイテック

●TX-80(定価￥108,000)…▶大特価￥ 79,000
(80MB、SCSI、SASI両対応)
●TX-130(定価￥138,000)…▶大特価￥ 99,000
(130MB、SCSI対応)
●TX-180(定価￥185,000)…▶大特価￥134,000
(180MB、SCSI対応)

夏休み限定フェア　パソコンラック〈送料無料〉

Ⓐ5段キャスター付
スライド式キーボード台
●1150(H)×640(W)×600(D)
定価￥38,000
特価(~~￥15,000~~)
￥14,000

Ⓑ4段キャスター付
●1250(H)×640(W)×700(D)
定価￥29,800
特価(~~￥11,000~~)
￥10,000

店頭新作ゲームソフト25～30%OFF!!ビジネスソフト25%より特価中

★通信販売お申込みのご案内★

〒144 東京都大田区蒲田4-6-7 TEL:03-3730-6271

お申込みはお電話でお願いします。お客様の〈住所〉〈氏名〉〈電話番号〉及び〈商品名〉をお知らせ下さい。●入金確認後ただちに商品をご送付いたします。

現金一括払い
銀行振込:お近くの銀行より(電信扱い)にてお振込み下さい。
現金書留:封筒の中に住所・氏名・商品名をご記入の上当社までお送り下さい。

クレジット
専用お申込用紙をお送り致しますので、必要事項をご記入、ご捺印の上ご返送下さい。手続きは簡単です。

オクト ラクラク クレジット表

3回	3.5	6回	4.5	10回	6.0	12回	6.0
15回	9.0	18回	11.0	20回	12.0	24回	12.5
30回	17.0	36回	17.5	48回	23.0	60回	33.0

振込先
富士銀行 久ヶ原(クガハラ)支店 ㊟No.1824
三菱銀行 蒲田支店 ㊟No.0278691
株式会社 億人(オクト)

※掲載の価格は変動しますので、まずは、お電話にてご確認ください。
※上記料金には、消費税は含まれておりません。消費税が付加されますので、詳しくは電話でお問合せ下さい。
※銀行振込、または、現金書留でご注文の際には、あらかじめ電話でご確認の上、お申し込み下さい。

超低金利クレジットOK

またまた秋葉原でおなじみの

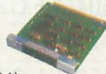
注目!!
夏のボーナス一括払い
手数料(金利)無料
(平成3年8月末をご利用下さい。)

7/15~8/15

●お近くの方はお
●本体単品で特
●ビジネスソフト定

X6800用ハードディスク(80M)
■TX-80(アイテック) SASI SCSI 両用
定価¥108,000 特価¥80,000
(送料・消費税込み¥83,430)

Fine Scanner-X68
(HAL研究所)X68000専用
■HGS-68(定価¥39,800)
特価¥26,000
(送料・消費税込み¥27,295)

X68000シリーズ専用 特価¥14,300
MIDIインターフェースボード
SX-68M(サコム)
(純生コンパチ)定価¥19,800
(送料・消費税込み¥15,244)

X68000メモリボード(シャープ&I/O・DATA)(送料¥500)
①CZ-6 BE1(600C用)定価¥35,000 (送料・消費税込¥27,295)…特価¥26,000
②PIO-6BE1-A 定価¥25,000 (送料・消費税込¥17,819)…特価¥16,800
③PIO-6BE2-2M 定価¥50,000 (送料・消費税込¥34,505)…特価¥33,000
④PIO-6BE4-4M 定価¥88,000 (送料・消費税込¥58,710)…特価¥56,500

ジョイスティック 送料¥500
●X-1PRO
定価¥9,500▶特価¥7,800
●ASCII STICK
定価¥6,800▶特価¥5,500

P&A超低金利クレジットをご利用ください!!

X68000-XVI 新発売!! (送料・消費税込み)

★先着100名様へ。
ゲームソフト(V-BALL¥7,900)をプレゼント!!

X68000-XVI▶セットでお買い上げの方に●ディスケット10枚●ジョイカード2ケプレゼント中!!

Ⓐセット:CZ-634C-TN+CZ-606D-TN…定価¥447,800▶特価価格はTEL下さい。

12回	?	24回	?	36回	?	48回	?	60回	?

Ⓑセット:CZ-634C-TN+CZ-613D-TN…定価¥503,000▶特価価格はTEL下さい。

12回	?	24回	?	36回	?	48回	?	60回	?

X68000-XVI-HD▶セットでお買い上げの方に●ディスケット10枚●ジョイカード2ケプレゼント中!!

Ⓐセット:CZ-644C-TN+CZ-606D-TN…定価¥597,800▶特価価格はTEL下さい。

12回	?	24回	?	36回	?	48回	?	60回	?

Ⓑセット:CZ-644C-TN+CZ-613D-TN…定価¥653,000▶特価価格はTEL下さい。

12回	?	24回	?	36回	?	48回	?	60回	?

※上記のモニターを、CZ-604D(定価¥94,800)、CZ-605D(定価¥115,000)、CU-21HD(定価¥148,000)に変更の場合、TEL下さい。超特価で販売致します。

X68000シリーズ~P&Aスペシャルセット=台数限定 送料、消費税込み

※セットでお買い上げの方に、●ディスケット10枚、●ジョイカード2個プレゼント中!!
先着100名様。ゲームソフト(V-BALL¥7,900)をプレゼント!!

SUPER

Ⓐセット:P&A特選セット
■CZ-604C (本体定価¥348,000)
+
■CZ-606D (モニター定価¥79,800)
▶P&A超特価 ¥297,000

Ⓑセット
■CZ-604C+CZ-604D
定価¥442,800………▶特価¥303,000
Ⓒセット
■CZ-604C+CZ-605D
定価¥463,000………▶特価¥321,000
Ⓓセット
■CZ-604C+CZ-613D
定価¥483,000………▶特価¥335,000
Ⓔセット
■CZ-604C+CU-21HD
定価¥496,000………▶特価¥344,000

PRO-II

Ⓐセット:P&A特選セット
■CZ-653C (本体定価¥285,000)
+
■CZ-606D (モニター定価¥79,800)
▶P&A超特価 TEL下さい!!

Ⓑセット
■CZ-653C+CZ-604D
定価¥379,800………▶特価¥250,000
Ⓒセット
■CZ-653C+CZ-605D
定価¥400,000………▶特価¥269,000
Ⓓセット
■CZ-653C+CZ-613D
定価¥420,000………▶特価 TEL下さい!!
Ⓔセット
■CZ-653C+CU-21HD
定価¥433,000………▶特価¥290,000

SUPER-HD

Ⓐセット:P&A厳選セット
■CZ-623C (本体価格¥498,000)
+
■CZ-606D (モニター定価¥79,800)
▶P&A超特価 ¥378,000

Ⓑセット
■CZ-623C+CZ-604D
定価¥592,800………▶特価¥384,000
Ⓒセット
■CZ-623C+CZ-605D
定価¥613,000………▶特価¥403,000
Ⓓセット
■CZ-623C+CZ-613D
定価¥633,000………▶特価¥415,000
Ⓔセット
■CZ-623C+CU-21HD
定価¥646,000………▶特価¥425,000

EXPERII

Ⓐセット:P&A厳選セット
■CZ-603C (本体価格¥338,000)
+
■CZ-606D (モニター定価¥79,800)
▶P&A超特価 ¥274,000

Ⓑセット
■CZ-603C+CZ-604D
定価¥432,800………▶特価¥280,000
Ⓒセット
■CZ-603C+DZ-605D
定価¥453,000………▶特価¥298,000
Ⓓセット
■CZ-603C+CZ-613D
定価¥473,000………▶特価¥313,000
Ⓔセット
■CZ-603C+CU-21HD
定価¥486,000………▶特価¥321,000

●本広告の掲載の商品の価格については、消費税は含まれておりません。●営業時間=平日AM10:00~PM7:00、日祭AM10:00~PM6:00

回~84回払いまでOK.!!　★頭金なし!★即日発送

P&Aがズバリ超特価セールでご奉仕!!

●価格は流通事情により変動致しますので、銀行振込・書留等の送付前に、あらかじめお電話にてご確認下さい。

立寄り下さい。専門係員が説明いたします。
価で受付します。詳しくは電話にてお問合せ下さい。
価の20%引きOK! TELください。

全国通販

超特価でクレジットが組める!!

X68000用ソフトコーナー(送料1ヶ~5ヶまで¥500)

品名	定価	特価
Z's STAFF PRO68K Ver.2.0(ツァイト)	¥58,000	¥38,800
Z's TRIPHONY デジタルクラフト(ツァイト)	¥39,800	¥27,800
テラッツォ(ハミングバード)	¥19,400	¥14,200
KAMIKAZE(サムシング・グッド)	¥68,000	¥44,800
C & Professional Pack(マイクロウェアジャパン)	¥58,000	¥41,000
Final Ver3.2(エーエスピー)	¥38,000	¥29,600
C-compiler PRO68K Ver.2 CZ-245L	¥44,800	¥33,300
CARD PRO68K CZ226BS	¥29,800	¥21,200
X.BAS to C CHECKER CZ-260LS	¥9,800	¥7,400
OS-9/X68000 CZ219SS	¥29,800	¥22,500
AI-68K CZ234LS	¥188,000	¥138,000
THE 福袋 V2.0 CZ224LS	¥9,900	¥7,400
SOUND PRO68K CZ-214MS	¥15,800	¥11,400
MUSIC PRO68K CZ213MS	¥18,800	¥13,400
Sampling PRO68K CD215MS	¥17,800	¥12,700
MUSIC-studio PRO68K CZ-252MS	¥15,800	¥21,400
MUSIC-PRO68K〔MIDI〕247MS	¥28,800	¥20,700
New-print Shop 221HS	¥19,800	¥15,500
Communication 223CS	¥19,800	¥14,200
Communication Ver.2 CZ-257CS	¥19,800	¥15,500
C-TRACE68 Ver.3.0(キャスト)	¥98,000	¥69,000
サイクロンEXPRESS α68	¥98,000	¥69,800
G68K Ver2 PRO	¥22,000	¥17,500
SX-WINDOW CZ-259SS	¥6,800	¥4,900
Gツール(ザインソフト)	¥28,000	¥18,900
たーみのる2(SPS)	¥17,800	¥13,300
マジックパレット(ミュージカルプラン)	¥19,800	¥14,500
Hyper word CZ-251BS	¥39,800	¥29,600

●ゲームソフト20%OFF OK!! (一部ソフト除く)

X68000用ハードディスク(送料¥1,000)

アイテック(SCSIタイプ)

■TX-130(130MB)…定価¥138,000▶特価¥102,000
(送料・消費税込み¥106,090)

■TX-180(180MB)…定価¥185,000▶特価¥136,000
(送料・消費税込み¥141,110)

プリンター(ケーブル・用紙付)　(送料¥1,000)

機種	定価	特価
■CZ-8PC5-BK NEW	¥96,800	特価価格はTEL.!!
■CZ-8PK10	¥97,800	¥71,000
■CZ-8PG2	¥160,000	特価価格はTEL.!!
■CZ-8PG1	¥130,000	特価価格はTEL.!!

周辺機器コーナー(送料¥500)

No.	品名	定価	特価
①	CZ-8NSI	¥188,000	¥145,000
②	CZ-6VTI	¥69,800	¥52,500
③	CZ-6TU	¥33,100	¥24,500
④	BF-68PRO	¥19,800	¥15,300
⑤	CZ-6BEI	¥35,000	¥26,000
⑥	CZ-6BEIA	¥38,000	¥28,600
⑦	CZ-6BE2A	¥59,800	¥44,200
⑧	CZ-6BE2B	¥54,800	¥40,800
⑨	CZ-6BFI	¥49,800	¥38,200
⑩	CZ-6BPI	¥79,800	¥60,000
⑪	CZ-6BMI	¥26,800	¥20,300
⑫	CZ-6EBI	¥88,000	¥66,500
⑬	AN-S100	¥36,600	¥28,500
⑭	CZ-6SDI	¥44,800	¥35,000
⑮	CZ-6BN1	¥29,800	¥22,600
⑯	CZ-6BV1	¥21,000	¥15,900
⑰	CZ-64H	¥120,000	¥91,500
⑱	CZ-6BG1	¥59,800	¥45,000
⑲	CZ-6BU1	¥39,800	¥30,300
⑳	CZ-6PVI	¥198,000	¥153,000
㉑	CZ-6BS1	¥29,800	¥22,300
㉒	CZ-8NJ2	¥23,800	¥18,500
㉓	CZ-6BL2	¥298,000	¥222,000
㉔	JX-100S	¥89,800	¥48,500
㉕	JX-220	¥146,000	¥107,900
㉖	IO-735X	¥248,000	¥169,000

モデムコーナー　(送料¥1,000)

■COMSTARZ CLUB24/5
(NEC) 定価¥39,800
特価¥26,500 (送料・消費税込み ¥28,325)

■MD-24FB5V
(オムロン) 定価¥39,800
特価¥27,400 (送料・消費税込み ¥29,252)

P&A特選パソコンラック　(送料無料)

①3段¥9,000　②4段¥10,800　③5段¥14,800

全機種=移動自由(キャスター付)・キーボード収納可(5段のみ)=1230(H)×600(D)×650(W)

中古パソコンはP&Aにおまかせ!!

その場で高価現金買取り・高価下取りOK!!

■まずはお電話下さい。
03-3651-1884、FAX:03-3651-0141

■下取り・買取りでお急ぎの方、直接当社に来店、または、宅急便にてお送り下さい。

●下取りの場合…………価格は常に変動していますので査定額をお電話で確認して下さい。(差額は、P&A超低金利クレジットをご利用下さい。)

●買取りの場合…………現品が着き次第、2日以内に買取り金額を連絡し、振込み、又は書留でお送り致します。

●近郊の方は、P&A本店まで、直接お持ち下さい。即金にて、¥1,000,000までお支払い致します。

《便利な超低金利クレジットをご利用下さい》

●月々¥1,000円からOK!!　●ボーナス払いOK(夏冬10回までOK)
●支払い回数 1回~84回　●お支払いは、8ヶ月先からでもOK!!

アフターサービス万全
全商品保証付、専門の担当者がお客様の立場で対応します。
初期不良、輸送トラブルetc.
万が一初期不良、輸送トラブルが発生しました際には、即交換させていただきます。

●定休日/毎週水曜日=第3水曜(祭日の場合は翌日になります)

通信販売お申し込みのご案内

〔現金一括でお申し込みの方〕
●商品名およびお客様の住所・氏名・電話番号をご記入の上、代金を当社まで、現金書留でお送りください。(プリンター・フロッピーの場合、本体使用機種名を明記のこと)

〔銀行振込でお申し込みの方〕
●銀行振込ご希望の方は必ずお振込みの前にお電話にてお客様のご住所・お名前・商品名等をお知らせください。
(電信扱いでお振込み下さい。)

〔振込先〕 住友銀行 新小岩支店
普通預金 1451576 ㈱ピー・アンド・エー

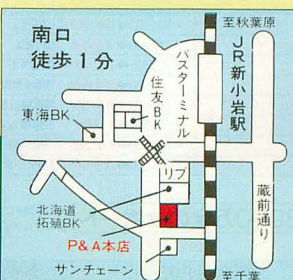
〔クレジットでお申し込みの方〕
●電話にてお申し込みください。クレジット申し込み用紙をお送りいたしますので、ご記入の上、当社までお送りください。
●現金特別価格でクレジットが利用できます。残金のみに金利がかかります。
●1回~84回払いまで出来ます。但し、1回のお支払い額は¥1000円以上。

超低金利クレジット率

回数	3	6	10	12	18	24	36	48	60	72	84
手数料	3.5	4.5	6.0	6.0	11.0	12.5	17.5	23.0	29.5	38.0	45.5

マイコン専門ショップ

株式会社ピー・アンド・エー
〒124 東京都葛飾区新小岩2丁目1番地19号
☎03-3651-0148(代)　FAX. 03-3651-0141

営業時間
平日:AM10:00~PM7:00
日祭:AM10:00~PM6:00

●現金書留及び銀行振込でお申し込みの方は、上記商品の料金に3%加算の上でお申し込み下さい。詳しくは、お電話でお問い合せ下さい。

AOYAMA
FOR THE EVOLUTION OF YOURLIFE

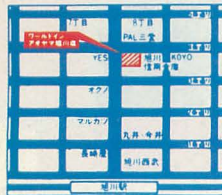
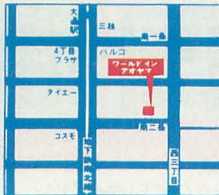
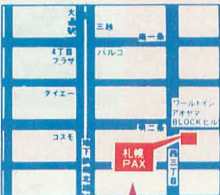
【各店のお休み】
7月のお休み／4日(木)・11日(木)・18日(木)・25日(木)
8月のお休み／8日(木)・14日(水)・15日(木)・16日(金)・22日(木)・29日(木)

ワールドインアオヤマにおまかせ下さい！

INFORMATION

電話でのご注文の場合
03-3987-7771
北海道受注センター 011-251-6771
九州受注センター 092-672-7771
お好きな時間にお電話を！

ファクシミリでご利用の場合
03-3985-5221
●ご注文方法(黒色のボールペン、またはサインペンでご記入下さい。)
(1)電話番号・住所・氏名又はお客様番号、お支払い方法をご記入下さい。

お客様相談室
03-3987-7795
すでにご注文いただいているお届け時間(時期)やメンテナンス、その他のお問い合せは上記へお電話下さい。

旭川店
旭川市4条8丁目ツジビル
■営業時間／11:00～19:00

札幌店
札幌市中央区南2条西3丁目
リンクエギビル3F
■営業時間／11:00～19:30

札幌PAX店
札幌市中央区南2条西2丁目
ブロックビル6F
■営業時間／11:00～19:30

池袋店ソフト店
豊島区東池袋1-28-6
パールシティビル2F
■営業時間／11:00～19:00

池袋本店
豊島区東池袋1-28-1
■営業時間／11:00～19:00

福岡店
福岡市中央区渡辺通り4-9-25
ユーテックプラザ3F・地下鉄天神駅下車3分
■営業時間／11:00～19:30

―新コース―

※下記のコースお買上のお客様にソフト2本(ダウンタウン熱血物語、熱血高校サッカー編)各8,800円をプレゼント致します。
● X68000 CZ-653C〔本体〕+CZ-606D〔カラーディスプレー〕………合計¥382,400➡¥232,900
● X68000 CZ-653C〔本体〕+CZ-613D〔カラーディスプレーテレビ〕………合計¥437,600➡¥272,800

【福岡店のお休み】
7月のお休み／3日(水)・10日(水)・17日(水)・24日(水)
8月のお休み／7日(水)・14日(水)・15日(木)・16日(金)・21日(水)・28日(水)

X68000 EXV X68000 Bコース
CZ-634C〔本体〕……¥368,000
CZ-606D〔ディスプレー〕……¥79,800
マウスパッド……サービス
書類リターン……サービス
住友3M5′2HDパック……¥9,000
定価合計¥456,800➡現金大特価
全店統一で安すぎて表示できません。
クレジット業界最低の金利を使ってクレジットでも。

X68000 EXV X68000 Cコース
CZ-634C〔本体〕……¥368,000
CZ-607D〔チューナー付ディスプレー〕……¥99,800
マウスパッド……サービス
書類リターン……サービス
住友3M5′2HDパック……¥9,000
定価合計¥476,800➡現金大特価
全店統一で安すぎて表示できません。
クレジット業界最低の金利を使ってクレジットでも。

X68000 EXV X68000 Aコース
CZ-634C〔本体〕……¥368,000
CZ-614D〔ノングレアディスプレーテレビ〕…¥135,000
マウスパッド……サービス
書類リターン……サービス
住友3M5′2HDパック……¥9,000
定価合計¥512,000➡現金大特価
全店統一で安すぎて表示できません。
クレジット業界最低の金利を使ってクレジットでも。

X68000 EXV X68000 Eコース
CZ-644C〔本体〕……¥518,000
CZ-607D〔チューナー付ディスプレー〕……¥99,800
マウスパッド……サービス
書類リターン……サービス
住友3M5′2HDパック……¥9,000
定価合計¥626,800➡現金大特価
全店統一で安すぎて表示できません。
クレジット業界最低の金利を使ってクレジットでも。

X68000お買上げのお客様へ
各コースで御希望ソフトは「サンダーブレード」「ダウンタウン熱血物語」「ニュージーランドストーリー」「沙羅曼蛇」「ツインビー」「フルスロットル」「パックマニア」「ビーチバレー」「アルカノイド」「熱血高校ドッチボール」のうちいずれかからお選び下さい。

X68000 EXV X68000 Gコース
CZ-644C〔本体〕……¥518,000
CZ-614D〔ノングレアディスプレーテレビ〕…¥135,000
マウスパッド……サービス
書類リターン……サービス
住友3M5′2HDパック……¥9,000
定価合計¥662,000➡現金大特価
全店統一で安すぎて表示できません。
クレジット業界最低の金利を使ってクレジットでも。

ステレオMIDI音源セット X68000 パーツセット2
CZ-6BM1〔MIDIボード〕……¥26,800
CM-32L〔ローランドMIDI音源〕……¥69,000
MA-12AV×2〔ボーズアンプ内蔵スピーカー〕¥28,000
MUSIC-PRO-MIDI〔ソフト音源〕……¥28,800
ソングライブラリ〔ソフト音源データ〕…¥8,800
定価合計 ¥161,400➡¥134,900

¥6,900×24回	ボなし 頭なし
¥12,800×12回	ボなし 頭なし

X68000 MIDIセット X68000
CM-32L〔FM音源〕……¥69,000
SX-68M〔インターフェイス〕……¥19,800
MU-1〔MUSIC SOFT〕……¥19,800
MA12AV〔アンプ付スピーカ〕×2……¥28,000
MU-1〔音楽データ集〕……¥6,800
定価合計 ¥143,400➡¥115,200

¥5,900×24回	ボなし 頭なし
¥4,100×36回	ボなし 頭なし

X68000をはじめソフト&周辺機器類は、当社池袋店・札幌店・旭川店・福岡店にて実演中です。各店X68000コーナーが常設されております。

X68000ソフト&周辺機器

品名	価格	品名	価格	品名	価格
SCSIボード(CZ-6BSI)	¥29,800➡現金特価			Communication PRO-68K	¥19,800➡現金特価
システムサコムMIDIボード(SX-68M)	¥19,800➡¥15,300	CZ-6TU	¥25,000➡現金特価	Stationary PRO-68K	¥14,800➡現金特価
ステレオアンプスピーカ(BOSS)	¥30,000➡¥24,700	オムロンMD-24FP4 II	¥38,800➡¥27,600		
オムロン MD-24FB5V	¥39,800➡¥29,800	オムロンMD-24FP5 II	¥42,800➡¥29,800	BUSINESS PRO-68K	¥68,000➡¥51,000
スモークドフィルター(D1415)	¥8,000➡¥6,800	ローランドMT-32	¥64,000➡¥54,400	NEW Printshop PRO-68K	¥19,800➡現金特価
インテリジェントコントローラ	¥23,800➡¥18,900	Hyperword	¥39,800➡現金特価		
		CYBERNOTE PRO68K	¥19,800➡現金特価		
マルチワープロ(CZ-225BS)	¥32,000➡現金特価	C compiler PRO-68K	¥44,800➡¥33,600	Musicstudio PRO-68K ver1.1	¥28,800➡¥21,600
				MUSIC PRO-68K〔MIDI〕	¥20,500➡現金特価
拡張I/Oボックス	¥36,600➡現金特価	Teleportion(CZ-258BS)	¥22,800➡現金特価	ソングライブラリ —101曲集—	¥8,800➡現金特価
アンプ内蔵スピーカーシステム	¥28,500➡現金特価			Sampling PRO-68K	¥12,500➡現金特価
		SX-WINDOW ver1.1	¥6,800➡現金特価	SOUND PRO-68K	¥15,800➡¥11,500

X68000シリーズ周辺機器

品名	価格	品名	価格	品名	価格
CZ-8NS1	¥188,000➡¥141,000	CZ-8PC5	¥94,800➡現金特価	I/Oデータ 2MB増設RAM	¥50,000➡¥36,500
CZ-6BN1	¥29,800➡現金特価	IO-735X	¥248,000➡¥169,000	I/Oデータ 4MB増設RAM	¥88,000➡¥64,000
CZ-6VT1	¥69,800➡¥52,400	CZ-8PK10	¥97,800➡現金特価	GP-IBボード	¥59,800➡現金特価
CZ-6BV1	¥21,000➡現金特価	1MB増設RAM(CZ-600C専用)	¥35,000➡¥28,000	増設用RS-232Cボード	¥49,800➡現金特価
CZ-6PV1	¥198,000➡¥148,500	1MB増設RAM	¥28,000➡¥22,400	ユニバーサルI/Oボード	¥39,800➡現金特価
CZ-8PC4	¥96,800➡¥69,800	2MB増設RAM	¥60,000➡現金特価	数値演算プロセッサ	¥61,000➡現金特価
CZ-8PG1	¥130,000➡¥97,500	4MB増設RAM	¥138,000➡¥107,000	FAXボード	¥79,800➡¥55,800
CZ-8PG2	¥160,000➡現金特価	I/Oデータ 1MB増設RAM	¥25,000➡¥18,000	MIDIボード	¥26,800➡¥20,300

X68000万全のサポート
AOYAMAにて購入のX68000は万一故障の場合でも全国どこでも出張サービスがうかがいます。万一の場合ワールドインアオヤマサポート係にお電話下さい。お客様のお名前と電話番号だけで手続きは完了。

組合せ自由	激安金利にキャンパスクレジット	ゆっくり、お支払いは8ヵ月先から
各コース以外の組合せもコースをベースに周辺を合せたセットお支払いたって御希望のパターンをお組みいたします さあ、ご相談もお見積りも受注センターもしくは各店へお気軽に	手続きカンタン、大学生の為の超低金利クレジット。20歳以上の学生の方は原則として保証人様には連絡いたしません。	クレジット業界最低の金利を有効に使って、支払いは最長8ヵ月後から始まるクレジットでも。

下記のコースお買上のお客様にソフト2本(ダウンタウン熱血物語、熱血高校サッカー編)各8,800円をプレゼント致します。
● X68000 CZ-604C〔本体〕+CZ-606D〔カラーディスプレー〕……合計¥445,400➡¥298,000
● X68000 CZ-604C〔本体〕+CZ-613D〔カラーディスプレーテレビ〕合計¥460,400➡¥336,000
● X68000 CZ-623C〔本体〕+CZ-606D〔カラーディスプレー〕……合計¥595,400➡¥378,000
● X68000 CZ-623C〔本体〕+CZ-613D〔カラーディスプレーテレビ〕合計¥650,600➡¥409,000
● X68000 CZ-653C〔本体〕+CZ-606D〔カラーディスプレー〕……合計¥382,400➡¥232,900
● X68000 CZ-653C〔本体〕+CZ-607D〔カラーディスプレーテレビ〕合計¥402,400➡現金特価
● X68000 CZ-653C〔本体〕+CZ-613D〔カラーディスプレーテレビ〕合計¥437,600➡¥272,800
● X68000 CZ-653C〔本体〕+CU-21HD〔21型テレビ〕……合計¥450,600➡¥279,600

TELEPORTION+MD24FB5V定価合計 ¥62,600➡¥46,000

HGS-68〔68000用スキャナー〕 定価¥39,800➡特価¥29,800	P10-6BE1A〔1MB増設RAM〕 定価¥25,000➡特価¥19,800	68000外付40Mハードディスク ※注1 定価¥99,800➡特価¥63,000
TX-80(80MBHDD) ※注2 定価¥108,000➡特価¥88,000	TX-130(130MBHDD) ※注2 定価¥138,000➡特価¥110,000	TX-180(180MBHDD) ※注2 定価¥185,000➡特価¥148,000

※注1.SUPER・XVIには使用出来ません。注2.SUPER・XVI以外の使用時CZ-6BS1が必要です。

コースオプション特選品	CS-10〔ステレオマイクロモニター〕¥17,000➡¥14,400
PC-200〔キーボード〕……¥36,000➡特価	CP-40〔はなうた君〕……¥33,000➡特価

セット	価格
MIDIセット〔SX68M+CM32L+パロディウス〕	定価合計¥97,600➡¥79,800
ゲームセット〔ドラッケン+プリンスオブペルシャ+シグナトリー〕	定価合計¥30,500➡¥23,900
通信セット〔コミニュケーションPRO68K+MD24FP4 II〕	定価合計¥58,600➡¥42,000
通信セット〔X Link PRO68K+MD24FP5 II〕	定価合計¥62,600➡¥44,200

★X68000をコースにてお買い上げいただきましたお客様にX68000専用パソコンラックを¥6,900にてお届けいたします。

買ったお客様でしかわからないこのサービス！——ぜったいX68000を買うならアオヤマがオトク
●以前当社にてX68000及びX-1を御購入いただいたお客様に限り、CZ-8PC5〔定94,800〕を大特価にてお届けいたします。会員の方は会員ダイアルにてCall！
●X68000をセットでお買い上げいただいたお客様に限り、XE-1ST2を特価¥4,900またCTRACEを特価¥20,000にてCZ-8NJ2〔インテリジェントコントローラ〕〔¥23,800〕を特価¥16,900にてお届けいたします。御注文の際に合わせてお申し込み下さい。

信頼と安心のTSUKUMO

ツクモ全店

カツとび 7/20㈯〜7/31㈬迄 決算セール / 8/2㈮〜8/20㈫迄 夏ツクモ・ザ・バーゲン

X68000 大好評発売中！

X68000 XVI 快速16MHz

- CPUクロック周波数スピードアップ(16MHz)
- 増設メモリ本体内蔵可能(8MBまで)
- NEW SX-WINDOW搭載

■X68000XVI(CZ-634C-TN) 標準タイプ……………定価￥368,000

■X68000XVI-HD(CZ-644C-TN) HD内蔵タイプ……定価￥518,000

(買い換え・下取りも取り扱っております。是非、お尋ね下さい。)

お買得セット

X68000PROⅡセット

- CZ-653C(CPU) ● 1MB増設メモリー
- CZ-605D(モニター) ● 内蔵40MBハードディスク

決算特価￥395,000(消費税別途￥11,850)

クレジット例(42回払・税込) 初回￥14,465＋月々￥12,200×41回

X68000SUPERセット 20台限り

- CZ-604C-TN ……………￥348,000
- CZ-613D-BK ……………￥135,000

合計定価￥483,000

ツクモ決算特価

クレジット例(24回払・税込) 初回￥21,929＋月々￥19,900×23回

お買得メモリー

一流メーカー増設メモリーボード

1MB増設RAMボード (ACE/PRO/PROⅡシリーズ用) 決算特価￥17,500(消費税別途￥525)

2MB増設RAMボード 決算特価￥34,800(消費税別途￥1,044)

4MB増設RAMボード 決算特価￥61,500(消費税別途￥1,845)

※計測技研のメモリーボードも取扱っておりますので、価格についてはお尋ね下さい。

冬のボーナス一括払い8/1より受付開始!! 詳しくは☎03(3251)9911へ

X68000 ユーザー必須のコンピュータミュージック特別セット

Aセット
- CM-32L ……………￥69,000
- SX-68M ……………￥19,800
- Musicstudio Mu-1 Ver1.4 ……￥19,800

合計定価￥108,600

決算特価￥88,000(消費税別途￥2,640)

クレジット例(18回払・税込) 初回￥7,223＋月々￥5,600×17回

Bセット
- CM-64 ……………￥129,000
- SX-68M ……………￥19,800
- Musicstudio Mu-1 Ver1.4 ……￥19,800

合計定価￥168,600

決算特価￥138,000(消費税別途￥4,140)

クレジット例(24回払・税込) 初回￥7,603＋月々￥6,900×23回

マニアセット NEWセット
- SC-55(ローランドサウンドキャンバス)…￥69,000
- SX-68M ……………￥19,800
- Mu-1 SUPER ……………￥39,800

合計定価￥128,600

決算特価￥99,000(消費税別途￥2,970)

SUPERマニアセット NEWセット
- CM-64 ……………￥129,000
- SX-68M ……………￥19,800
- Mu-1 SUPER ……………￥39,800

合計定価￥188,600

決算特価￥154,000(消費税別途￥4,620)

Aセット と Bセット の場合、「Musicstudio PRO68K Ver2.0」又は、「Music PRO68K」<MIDI>のソフトの場合には￥9,500プラスになります。また、これらのソフトウェアがバージョンアップにより価格が変更になった場合には変更となります。

ローランド 追加オプション機器

ステレオマイクロモニター CS-10 ……定価￥17,000

MIDIキーボードコントローラー PC-200 ……定価￥36,000

はなうたくん CP-40 ……定価￥33,000

7号店2F(MIDI専門フロア)にもございます。☎03(3253)4199 担当・佐々木

人気爆発 X68000用TSドライブ

好評につき在庫お問い合せ下さい。

TS-3XR1 定価￥44,800

3.5インチフロッピーディスクドライブ

- 1ドライブタイプ ● 3.5インチ2DD/2HD対応ドライブ使用 ● ユーティリティソフト付属。(ディバイスドライバ)

決算特価￥35,800(消費税別途￥1,074)

新発売

X68000用ハードディスク

大容量記憶装置

TX-80 定価￥108,000 80MB SCSI/SASI両対応 決算特価￥88,000(消費税別途￥2,640)

TX-130 定価￥138,000 130MB SCSI対応 決算特価￥110,000(消費税別途￥3,300)

TX-180 定価￥185,000 180MB SCSI対応 決算特価￥148,000(消費税別途￥4,440)

※SCSIハードディスクとしてご使用の場合、本体がSUPER/XVI以外の場合にはSCSIボード(CZ-6BS1)が必要です。

(カラー：ブラック／グレー)

ビジネスツール
- ■Hyper WORD ……定価￥39,800
- ■Multiword NEW ……定価￥32,000
- ■FIXER Ver4.0 …… 決算特価￥15,800
- ■CARD PRO-68K Ver2.0 NEW 定価￥29,800

アートツール(ハード)
- ■JX-220X A4サイズカラーイメージスキャナー ……定価￥168,000
- ■ファインスキャナーX68 HGS-68 …… 決算特価￥31,800
- ■CZ-6VT1 カラーイメージユニット 定価￥69,800
- ■CZ-6BV1 ビデオボード ……定価￥21,000
- ■CZ-8PC5 48ドットカラー漢字熱転写プリンター NEW ……定価￥96,800

アートツール(ソフト)
- ■CANVAS PRO-68K ……定価￥29,800
- ■Z's STAFF PRO-68K Ver2 …… 決算特価￥46,400
- ■マジックパレット…… 決算特価￥15,800

開発ツール
- ■C Compiler PRO-68K Ver2.0 定価￥44,800
- ■XBAS TO C CHECKER PRO-68K 定価￥9,800

通信ソフト
- ■た〜みのる 2 …… 決算特価￥14,200
- ■Telepotion PRO-68K ……定価￥22,800

安心 迅速 高額 買い取りのツクモニューセンター店

ツクモ買い取りセンター

好評買い取り中！

電話受付(03)3251-9977(AM11:00〜PM5:00)

FAX受付(03)3251-0299(24時間)

この他電子手帳も大特価でお取り扱いしています。

ツクモグローバルカード

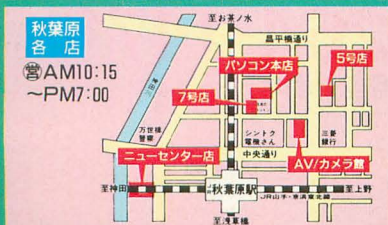
好/評/入/会/者/受/付/中/!!

〜国内・海外でも使える多機能

ジャックス・VISAの提携カードです。分割払い、ボーナス払いもOK！海外旅行傷害保険や各種サービスがついています。パソコン本店にあるキャッシングマシンでキャッシングOK！クレジット申し込みと同時にカード申し込みOK！ご入会希望の方は☎03(3251)9898又は各店で

★各店頭では、JCB・日本信販・DC・セントラル・マスター、他各種カードも取り扱っております。

ツクモ通販センター フリーダイヤル 受注専門 **0120-377-999**

商品についてのお問い合せは各店店頭又は……☎03(3251)9911へ

ツクモは「スーパーX PRO SHOP」です。

PRO STAFF **ツクモ**

パソコン本店 荒井 / N・C店 福地

九十九電機株

〒101-91 東京都千代田区神田郵便局私書箱135号

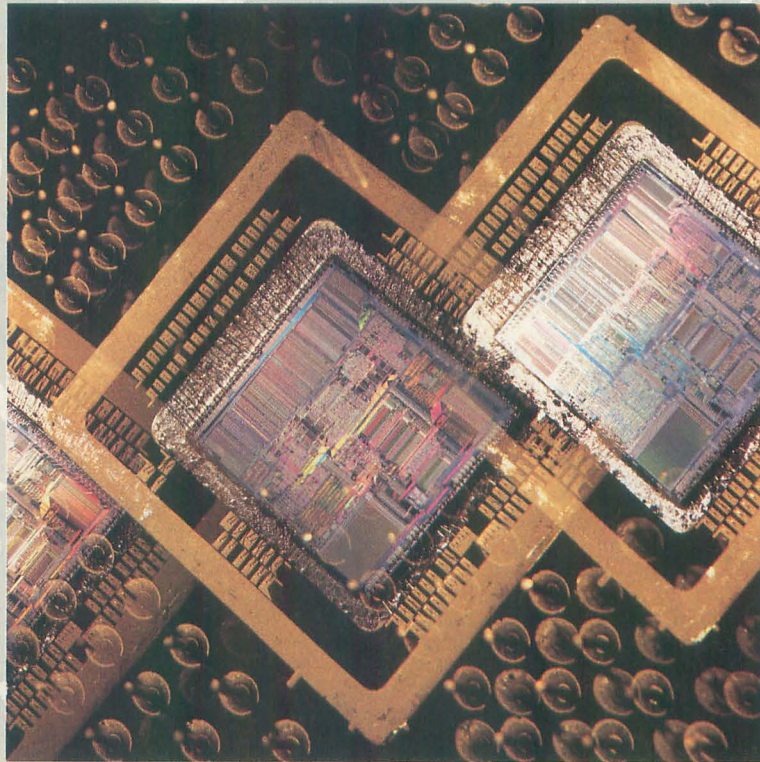
ツクモパソコン本店2F ☎03-3253-5599(担当／荒井) 休8/1,8,14,15

便利で安心な通信販売

ツクモ通販センター☎03-3251-9911

- ■ツクモニューセンター店 ☎03-3251-0987(担当／福地) 休8/1,8,14,15
- ■ツクモAV/カメラ館B1 ☎03-3254-3999(担当／川名) 休8/7,14,15
- ■ツクモ5号店 ☎03-3251-0531(担当／森) 休8/1,8,14,15
- ■名古屋1号店 ☎052-263-1655(担当／吉高) 休8/5
- ■名古屋2号店 ☎052-251-3399(担当／横山) 休8/8
- ■ツクモ札幌店 ☎011-241-2299(担当／田口) 休7/25,8/1,7,8

★表示価格には消費税は含まれておりません。 ★商品のご注文は在庫確認の上お願いします。

カード払い	全国代金引き換え配達	クレジット払い	現金書留払い	銀行振込払い	各種リース払い
通信販売での御利用カード、ツクモグローバルカード、VIPカード、セントラル、ジャックス※御本人様より電話で通信販売部へお申し込み下さい。	お申し込みは☎03-3251-9911へお電話1本！配達日の指定もできます。	月々￥3,000以上の均等払いも頭金なし、夏・冬ボーナス2回払いも受付中！	〒101-91東京都千代田区神田郵便局私書箱135号 ツクモ通販センター Oh!X係	事前に☎でお届け先をご連絡下さい。三和銀行 秋葉原支店(普)1009939 ツクモデンキ	くわしくは各店にお問い合せ下さい。ケースに合わせてご相談にのります！

SOFTWARE INFORMATION

今月の目玉はなんといっても「イース」。移植されるらしいという噂が流れてから，かなり時間がかかって，やっと7月19日に発売されることが決定した。「生中継68」もそろそろ発売される予定だから，熱い夏になりそうだ。

イース

イースといえば，いまさら説明の必要もないほど有名なアクティブロールプレイングゲーム。練り上げられたシナリオと適度なゲームバランスは評価が高く，いまでもファンが多い。当然X68000への移植を望む声も多かった。

そして，ついに「イース」のシリーズ第1作目が電波新聞社の手によりリメイクされて登場。グラフィックは全面的に手を入れられ，見てのとおりリアルなタッチの画面に変更されている。音楽もX68000オリジナルアレンジ。

2人の女神と6人の神官によって治められていた王国イース。しかし，クレリアという金属の使用が同時に怪物を呼び覚まし，王国は天空に逃れていった。そして数百年後，冒険家アドル・クリスティンはこの地に足を踏み入れ，魔物と神官たちの戦いに巻き込まれていく。

X1版などとはだいぶ印象が違うので，意見が分かれるかもしれないけれど，そのまま移植が多いなかでこのがんばりは評価したい。（浦）

X68000用　5"2HD版2枚組　9,600円(税別)
電波新聞社　☎03(3445)6111

発売前ソフトの動きに注目だ！

	タイトル	(前回順位)
1.	パロディウスだ！	1→
2.	ファランクス	3↑
3.	遥かなるオーガスタ	2↓
4.	生中継68	ー再
5.	A列車で行こうIII	4↓
6.	イース	6→
7.	サイレントメビウス	ー再
8.	キャンペーン版大戦略II	ー再
9.	マジカルショット	ー初
10.	キャメルトライ	ー初

ハーイ。TOP10のお時間デース。今月もさっそくチャートを見てみましョウ。

「パロディウスだ！」はまだまだ強いですね。移植完成度，ネームバリュー，ゲーム性と3拍子揃っただけあって首位を依然独走中。

その「パロディウスだ！」を追いかける2本の間で動きがありました。ファランクスがオーガスタを追い抜いています。画面作りや演出の勝利でしょうか。

「A列車で行こうIII」はワンランクダウン。X68000版はコンストラクションやマップ集も同時発売されているから，こんなに早く飽きちゃうはずないんですが。

ついにその姿を現わした「イース」。今月も6位にランクイン。なんでも製作スタッフがやたら「イース」に入れ込んでいて，満足いくまで手放さなかったからこんなに時間がかかったんだとか。X1版をすでにやっている人，X68000が初めて買ったパソコンの人の両方からまんべんなく支持を得ているようです。

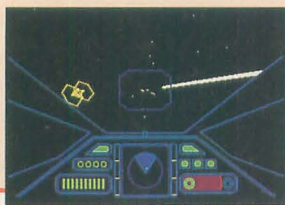
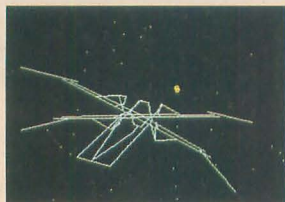
7位，「サイレントメビウス」は発売されて再登場。主な推薦理由は「ガイナックスが出したから」「マンガのファンだった」「アドベンチャーとしてもよくできていると思う」。

9位と10位は初めて名前を見る作品がふたつ。9位「マジカルショット」はM.N.M.ソフト初の入賞です。「こんなにいいゲームはヒットさせないとかわいそう」など，なかなか入れ込んだハガキが多いですね。健闘を期待しましょう。

さて，ではまた来月お会いしましョウ。お相手は，クララ・モルガンの影響を受けた私，浦川デシタ。（浦）

3D 2(仮称)

驚くなかれ，なんと有名SF映画の「とある」星への突入シーンのゲーム化だ。お馴染みのシーンも挿入されていて，臨場感たっぷり。視点切り替え，リプレイ機能装備も非常にうれしい。SIONでこのテのゲームのトリコになった人，そして，まだワイヤーフレーム3Dシューティングのスピード感や感情移入度の高さといった楽しさをまだ知らない人にも，この作品は絶対お勧め。近日堂々公開予定。キャッチコピーは,「ああ，思わず体が動いてしまう」。(R.A.)

X68000用　5″2HD版　価格未定
M.N.M.Software　☎0423(60)3084

F15ストライクイーグルII

「GUNSHIP」に続いて，マイクロプローズジャパンから発売されるのが，この「F15ストライクイーグルII」。もうおわかりのとおり，フライトシミュレータだ。フライトシミュレータはX68000には少ないだけに期待されるところ。

このゲームはすでにPC-9800では発売中なので，どこかで見かけた人も多いと思う。ポリゴンで表現されたわりとにぎやかな地形の中を，細かくモデリングされた戦闘機が飛び回る。もちろん，ミッションもいろいろなタイプのものが多数用意されている。

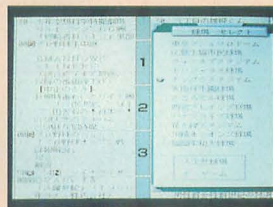
飛び回れるのはリビア，ペルシャ湾，ベトナム，中近東の4つの地域。そして，その地域によってさまざまな敵戦闘機が登場する。視点切り替えもばっちりできるし，オートパイロット機能もついている。掲載した写真はまだ開発途中の画面だが，X68000らしいグラフィックに仕上がっているようだ。(R.A.)

X68000用　5″2HD版　価格未定
マイクロプローズジャパン　☎0423(33)7781

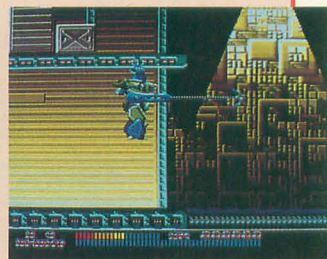
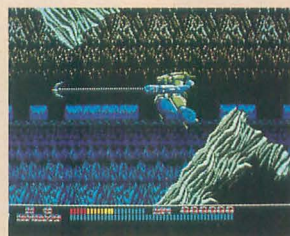
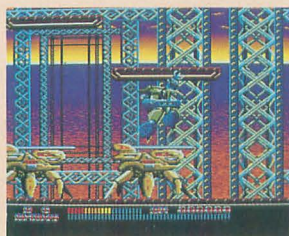
生中継68

今月は「生中継68」の演出の部分を詳しく紹介しよう。試合前にはチームや球場などを選択するのだが，写真のとおり，なんと選んだ組み合わせ，球場が新聞に記入されるのだ。うーん，うまい演出。そして，試合後はご存じプロ野球のニュースが放送される。ここで登場するキャスターは試合前に十橋さんなど，お馴染みのメンバーのなかから選択。試合のポイントなどを紹介してくれる。そして，サウンド。試合中ひっきりなしに流れている，さまざまな効果音。「1番，センター，背番号2」などという球場内のアナウンス，状況に応じての観客の歓声。もちろん，打ったり捕ったりというときの音もリアルなものが用意されている。音楽もゲームにマッチした，カッコイイものを多数収録。

ところで，このゲームに登場する「KBCテレビ」という放送局は九州に実際にあるような気がする……。(R.A.)

X68000用　5″2HD版2枚組　9,800円(税別)
コナミ　エンタテイメント　☎03(3264)5678

アクアレス

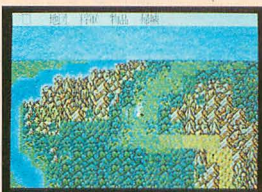
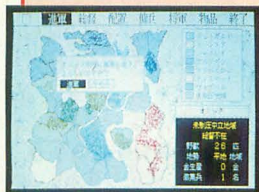
エグザクトの新作，「アクアレス」のゲーム内容がわかったのでお伝えしましょう。ロボットを操り敵を破壊していく正統派ロボットバトルに，ジャンプとワイヤーによる高度なアクション性を含んだ，新しいタイプのゲームになりそうです。特徴であるワイヤーアクションは，空中でジャンプボタンを押して発射されるワイヤーを地形にひっかけることで，多種多様な移動を実現することができます。そのため自機の動きはどことなく軽快な印象を与えてくれるほどですが，そのなかで敵と激しいバトルがくりひろげられるわけですから，どうやら熱いゲームになるのは間違いないといえるでしょう。硬派アクションの王道まっしぐらという感じで，力作を作っている熱意が伝わってくるようです。すでにナイアスでお馴染みの多重背景も，お約束のように今回もバキバキメリメリの具合にミガキがかかって暴れていますので，技術のエグザクトの本領発揮，好感の持てる期待株といえる一作です。(八)

X68000用　5″2HD版2枚組　8,700円(税別)
エグザクト　☎025(247)9160

ライヒスリッター

ファミコンなどでも有名なエニックスが一風変わったRPGを開発した。その名もライヒスリッター。パッケージにもロールプレイング・シミュレーションゲームと書かれているとおり，シミュレーションの要素を多分に含んでいるのは見逃せないポイント。どちらかといえば三國志などの歴史シミュレーションゲームに探索モードや魔法，ヒットポイント制を加えた感じ，といったほうが近いのではないだろうか。

主人公はシュトガルド国の国王となり，折しも復活を果たした魔王を倒すために，大陸を統一していかなければならない。最終的な目標に向かうのは主人公ひとりでも可能だが，頼りにできるのは将軍という存在。彼らなしに統一を達成するのは難しいかもしれない。戦闘シーンでは陣形を選択しなければならないのも，シミュレーションっぽい。

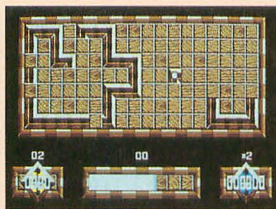
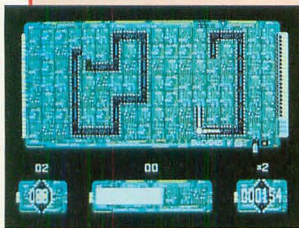
システムの新鮮さやシナリオなどはがんばっているのだが，スクロールや表示の切り替えなどが遅いことや，PC-9801からのベタ移植なのが気にかかる。　(S.K.)

X68000用　5"2HD版3枚組　8,000円(税別)
エニックス　☎03(5272)2374

ループス

「プリンス・オブ・ペルシャ」のブロダーバンドジャパンの次なるゲームはこれ，アクションパズルの「ループス」だ。

画面上にひょいひょいと現れるピースを好きなところに置いていく。そのピースをつなげて輪を作ると，ピース全体が消えて得点になるのだ。複雑な形をした長いループを作るほど高得点が入るんだけれど，次にどんな形のピースが来るかわからないし，しだいにややこしい形のピースが増えてきて，プレイヤーをじゃましてくれる。ピースを制限時間内に3回置けなかったら"GAME OVER"。長くプレイを続けるためには，どんなピースにも対応できる柔軟な発想と，素早い指さばきが必要なのだ。

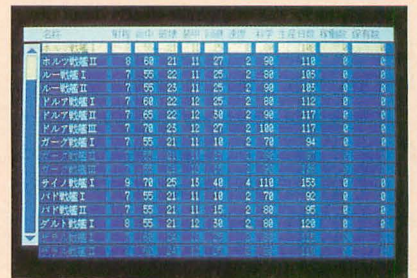
ゲームには7種類のグラフィックモードが用意され，難易度も9段階に選択可能。2人同時プレイもできるし，押さえるところはきちっと押さえてあるゲームだといえるだろう。　(浦)

X68000用　5"2HD版　7,800円(税別)
ブロダーバンドジャパン　☎03(3341)1135

インペリアルフォース

これからどんどんと登場するシステムソフトの新作。そのうちのひとつがこのSFシミュレーションゲームだ。プレイヤーは8種類の宇宙人からひとつを選択。宇宙人によって，生産する軍艦の特徴が異なるのがポイント。銀河に散在する惑星を探しだし，占領するのが目的だ。

また，最初はどんな惑星がどこにあるかわからないので，自分で偵察部隊を派遣して惑星を探さなければならない。近距離にある惑星同士はネットワークで結ばれ，交流によって惑星の技術力や生産力が向上するようになっている。

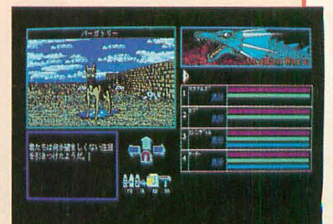
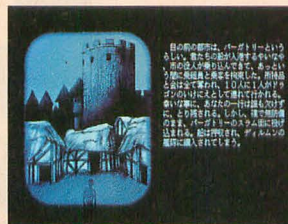
シミュレーションというと「大戦略III」のようなヘビーなものばかり思い浮かべてしまうかもしれないが，このゲームでは艦船の種類や惑星のパラメータなどをスッキリまとめている。マップもゲームごとに新しいものが作られるし，コンピュータが考えている間プレイヤーが待たされることもないので，気軽に遊べるタイプのシミュレーションといえそうだ。　(浦)

X68000用　5"2HD版　価格未定
システムソフト　☎092(752)5278

ドラゴンウォーズ

バトルチェスを送り出したスタッフが作り上げたファンタジーRPG，それがこの「ドラゴンウォーズ」だ。

かつては楽園とうたわれたディルムン諸島にやってきた主人公たち。しかし，島に着くやいなや身ぐるみはがされて，スラム街に放り込まれてしまう。邪悪な魔術師ナムターが王様に取り入って，この王国を支配していることを知った彼ら。ディルムンの平和を取り戻すため，そして，自分たちが生き延びるため，地獄から這い出てきた獣，ナムターを倒すことを決意するのだった。

ゲームにはスキル制度が取り入れられている。そして，オートマッピング機能も装備。経験値を稼いで楽しむ戦闘型RPGというよりは，シナリオ重視型RPGだといえそうだ。次々と起こるイベントを切り抜ける方法はひととおりではなく，人によって違った展開が楽しめるようになっているのも魅力のひとつ。アメリカ生まれの強力ファンタジーRPGだぞ。　(浦)

X68000用　5"2HD版3枚組　9,800円(税別)
スタークラフト　☎03(3988)2988

殺意は潮風に乗って

Nishikawa Zenji
西川 善司

「琥珀色の遺言」に続く，藤堂龍之介探偵日記シリーズ第２弾がこの「黄金の羅針盤」だ。甲板に置かれた樽の中から発見された白骨死体。それが大海原を行く豪華客船“翔洋丸”の安らかな航海を乱す事件の始まりであった。

「藤堂さん，お仕事は何をなさってますの？」

「私の仕事は人生の謎を探ることです」

「えっ？　それはどういうことですの」

「私は探偵という仕事をしています」

私は藤堂龍之介，私立探偵。400年前にモーリシャス島より全滅した，飛べない鳥とは何の関係もない。

黄金の羅針盤◆◆◆◆◆◆◆◆◆◆◆◆◆◆

1923年9月9日，亜細亜汽船が誇る豪華客船翔洋丸は桑港(サンフランシスコ)を横浜へ向けて出航した。

「洋上を翔ぶように駆けよ」。建造当時，こうした思いを込めて名づけられた翔洋丸は，日本を代表する豪華客船であった。が，船齢15年となった現在，あいついで建造される新型船に隠れ，華やかな表舞台から遠ざかりつつあった……。

事件というものは何の前触れもなく起こるものだ。

出航後２日目，プロムナードデッキを急ぎ足で歩いてきた乗客，一条菊子と，ボーイ，尾崎康平がぶつかった。そのどちらかの体が隅に置かれていた樽に当たったのか，とにかくなんらかのショックで樽の中身が甲板へとさらけ出された。

その場に居合わせた青沢夫妻の血の気をも奪った樽の中身。それは……まぎれもなく人間の白骨化した死体だったのである。

X68000用　5"2HD版3枚組　9,800円(税別)
リバーヒルソフト　☎092(771)3217

こうして，それまで永遠に続くかと思われた安らかな航海に終止符が打たれ，得体の知れない殺意の羅針盤が誰かを指し始めたのである……。

アースの間◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆

その晩，このことを一条菊子に打ち明けられた私は，探偵としての探求心と人間としての好奇心が強く刺激されるのを感じた。その白骨死体は指輪をしており，ボーイの知らせであとから駆けつけた船長と司厨長の赤松は，その場にいた４人に堅く口止めしたという。何かが起こる……，そんな予感が私の胸を通りすぎた。

私はラウンジから部屋に戻り，ベッドの下から鞄を取り出して中身を確認した。

トランプ……パッケージを買うとついてくるものだ。特に事件とは関係がないものと思われる

事件解決報告書……事件解決後，これをリバーヒルソフトに送ると素敵なプレゼントが抽選で当たるらしい。楽しみだ

乗船券/乗船案内……豪華な作りだ。しかし，私個人の意見としてはもっとこういうものは質素でいいからソフト自体の値段を下げるべきだと思う

船内マップ……これは便利だ。誰がどの部屋にいるのかを書き込むこともできる

私は荷物を確認したあと，マウスの右ボタンを押してメニューを出し，「捜査分析」を選択，「手帳」を開いた。この手帳には写真や矢印なんかを張ったり書き込んだりして，複雑な人物関係を見やすく整理することができる。「鞄」ではアイテムの説明を読むことができる。最後の「海図」は事件の捜査進行状況を確認することができる。便利で都合のいいものだ。翔洋丸が横浜に近づけば近づくほど，事件が核心に迫っていることを表しているらしい。

私はひととおり持ち物を確認すると，再びベッドの下へとしまい，マウスで「出口」を選び，「白骨死体」の目撃者たちを尋ねるべく部屋を出た。

捜査初日◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆

「出航してから何か変わったことはありましたか」

「アリマシタ，とても大変なコトが」

「えっ，それは何ですか」

「あなたとであったコトデス。ワタシの人生にとって最高の出来事デス」

クララ・モルガンは熱い眼差しで私を見つめ，こういった。

証拠品を探し出し，容疑者を問いつめろ

ふだんはこのような画面

捜査1日目は，いうならば事件の関係者との顔合わせだ。その人物とはまったく関係のないような事実や人についても遠慮せず話題に出し，相手の顔色をよく観察することが大切だ。私の友人のひとりであるJ.B.ハロルドもそういっていた。

「……そ，それは」

などということをいうやつは「怪しい」。何か知っている，と。

さあ，船上のロマンスにふけっている場合ではない，さっさと次の人物を捜して話をしなければ。こうして船内を歩いていると妙なことに気がついた。それは「いい加減な顔」をした人物はどうも事件には無関係らしいということだ。これは大きな手掛かりだ。顔に**鼻がない**人物はそういう病気なのではなく，単なる脇役キャラということを表現しているようだ。

さて，「コロンボ」並みのしつこさであれこれ聞く私に，ほとんどの人物が怒ってどこかへ行ってしまった。問題の「白骨死体」についても何の情報も得られなかった。しかし，白骨死体がしていた指輪のスケッチを見て，麻生多加子の表情が変化したように思われた。あんな美しく清純そうな人が事件に関係しているのだろうか。私は心にわだかまりを残したまま，喫煙室のドアを開け外に出ようとした。その瞬間，目の前が真っ暗になりナレーターの声が時の経過を告げた。

捜査初日は関係者を怒らせるだけでよかったらしい。私は海図を開き翔洋丸が1センチほど横浜に近づいたのを確認してほっとした。

血相を変えてボーイが飛び込んできた

移動画面，行くべきところはたくさんある

殺意の旋律◆◆◆◆◆◆◆◆◆◆◆◆◆◆

「甲板の樽の中から白骨死体が発見されたという話を知っているかい」

「知らないね。でも，この船になら死体の入った樽のひとつやふたつ乗っていても，なんらおかしなことはないさ」

翔洋丸専属楽団のバイオリニスト，雨宮義人は酒臭い息をこもらせてそういった。どうもこの翔洋丸にはその美しい外観とは正反対に，ドロドロとした人間模様があるようだ。

＊　＊　＊

私はその日の夜，夕食時に私を捜していたという麻生多加子の部屋「ビーナスの間」を訪れた。「指輪」のことについて何か話してくれるのだろうか。黄銅の円窓の向こうでは漆黒の空に星が2,3揺れていた。甘い香りのする室内を見回していると，麻生多加子はついに話を切り出した。

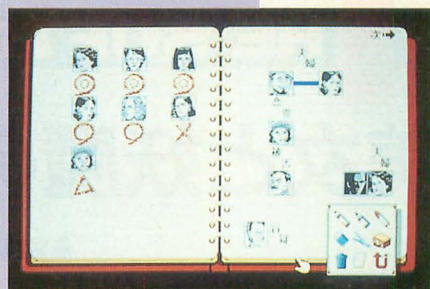
「藤堂さん，実は……」

と，そのとき，

「藤堂さん，た，大変です！　片桐事務長が殺されました」

ボーイの尾崎君がノックもせずに飛び込んできた。この世の終わりでも見たようなその表情は，私と麻生多加子を恐怖で満たすに必要十分であった。

「わかった，すぐ行く。君は船長と赤松さんを呼んできなさい」

「はい」

人物関係を手帳にうまく整理しよう

事件の進行状況はこの海図で確認できる

龍之介の予感◆◆◆◆◆◆◆◆◆◆◆◆◆

悪夢のような一夜が過ぎ，あの「白骨死体発見事件」もついに公表された。

今になって雨宮の言葉が妙に気になる。この翔洋丸にいかなる秘密が隠されているというのか。また，麻生多加子の秘密とは。そして……，あの「白骨死体発見事件」は片桐事務長の死体と何か関係があったのだろうか。いや，……ない。私の探偵としての勘が「No」を弾き出した。これはきっと単なる引き金にすぎなかったのではなかろうか。ということは，新たな惨劇もありうるのか。そして，白骨死体がしていたという指輪はいったい……？

次なる事件の発生の予感を胸に秘め，捜査の正式許可を得るべく，私は乗務員幹部が集う会議室の扉の前に立った。

翔洋丸は行く◆◆◆◆◆◆◆◆◆◆◆◆◆

ここまでで紹介したのは，ほんのゲームの触りの部分。これからどんどん人間関係が浮き彫りにされていき犯人像が明らかになっていく。そして，最後にはなんとも意外な人物が××して……，××してしまうという……。ああ，危ない危ない，ばらしてしまうところだった。「黄金の羅針盤」をやらずして，1991年のアドベンチャーは語れない。さあ，立ち読みしている君はそのままソフト屋へ（にしてもOh!Xは買ってよ）。定期購読の人は時計を見て。まだ7時前なら間に合う，ソフト屋へ直行だ。ヨドバシならゴールドポイントカードは忘れんなよ。

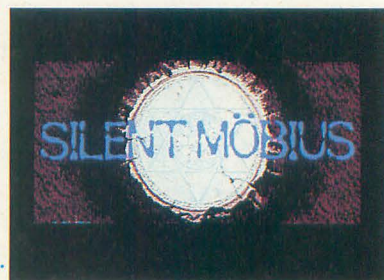
アレと比べてみる

非常に似通った設定のゲームにタケルの「ノスタルジア」がある。ゲームソフトにしちゃ，あっちもこっちもチト値段が高い。でも，両方ともとてもキャラクターやストーリーが面白く，グラフィックもよろしい。ここでちょっと比較をしてしまうと，話の面白さは両方とも互角，音楽も互角，値段は「ノスタルジア」のほうが高い，アドベンチャーとしては「黄金～」のほうが出来はいいが，小説としては「ノスタルジア」のほうが上か？　主人公の藤堂龍之介と山田カスケはともになかなかの魅力。いずれにせよ，どちらとも（善）は気に入ってしまった。そうそう，どっちともまだまだ続編が出るようだから楽しみだよね。

総合評価	0　5　10
シナリオ	★★★★★★★★★
増設RAM対応	★★★★★★★★★
スピード	★★★★★★★
グラフィック	★★★★★★★★★★
キザなせりふ	★★★★★★★★★★
船上のロマンス	★★★★★★★★

美女と妖魔の華麗な戦い

麻宮騎亜の人気コミック「サイレントメビウス」がアドベンチャーゲームになった。新たに書き下ろされたシナリオ，華麗なグラフィック，迫力溢れるサウンド。これはもう「サイレントメビウス」の世界にのめり込むしかない。

Komura Satoshi
古村 聡

私もX68000にも出ないかなと期待しつづけ，この半年間「ねえねえ，まだなのー？」とダダをこねては編集さんをせっつき，レビューの仕事がきてからは１日も長く遊んでられますようにと願い，締め切りを忘れ遊びまくってしまうという極悪非道の行いをやってしまいました。いいもん，こうなったら悪魔にでも妖魔にでも，魂売ってやるから。

お待たせしました。X68000版サイレントメビウスの登場なのです。このサイレントメビウスはコミックコンプに連載されている同名漫画が原作で，８月17日（この本の発売日の１カ月後だな）には劇場アニメが映画館で公開されるという，なかなかのヒット作品なのです。

で，このサイレントメビウス，なにがそんなに人気なのか。ひとつの理由は非常に背景，メカニック，妖魔のデザインが緻密で，「こりゃあ，いったい１ページあたりいくらお金がかかってるんだ？」というくらいトーンを貼りまくり，とにかく絵を見せていること。もうひとつは，登場人物である女の子の１人ひとりが個性的でかわいいこと。由貴ちゃんがいい，香津美がいいと，それぞれファンがついてしまっているおそろしさ。単行本の第３巻のゲストページにもレビアさんとラリー課長が好き，とか書いてる漫画家さんがいたし。

X68000用 5"2HD版7枚組 14,800円(税別)
ゼネラルプロダクツ ☎0422(22)1980

しかし，ファンがこういう感じで，はたしてこのゲームはすべてのサイレントメビウスファンを満足させられたんでしょうか。ゲームには出演しないキャラがいたりしたら，さぞかしファンは怒り狂うことでしょう……。

空に浮かぶ白い貴婦人◆◆◆◆◆◆◆◆

香津美，レビア，那魅，キディ，由貴の５人と私は，東京の空に浮かぶタイタニック号に着艦しようとしている。

2026年春のある日。東京の人々はその頭上に信じられないものを見た。頭上の雷雲のなかから巨大な船が現れたのである。軍や警察が出動したが，船を覆う巨大な結界のため近づくことができない。船窓に動く影からおそらく船内には人がいるに違いないこと，そして，その船は1912年に北大西洋で沈没した豪華客船タイタニック号にそっくりであることだけが確認された。

事態を憂慮した政府はケンブリッジ大学でタイタニック号を専門に研究している私と，AMPに出動を要請した。

常識と想像を絶するクリーチャーズトラップ，あるいは第３種事件と呼ばれる不可解な事件。これら妖魔によって引き起こされる第３種事件に対し創設されたのが，対妖魔用の特殊警察，“AMP”である。

「驚いたな。これは写真で見た，航海当時のタイタニックとまるで同じじゃないか」

それも，そっくりであったのは船の姿形だけではなかったのである。空の上にあるにもかかわらず，このタイタニック号の煙突からはもくもくと煙が立ち昇り，船員たちは黙々とデッキブラシで掃除を行い，展望台からは海が見え，甲板の向こうからはなんと潮の香りさえするのだ。

「おい，見ろよ。船員だ」

「驚いたな……。妖魔のやつ，ここまで完全に実体化させるとは……」

我々は船員に話しかけてみた。彼らは今日が４月14日だといっている。まさか？ ４月14日。そう，それこそはまさしく，1912年サウサンプトン港から処女航海に出港，氷山に衝突して1517名の乗客とともに豪華客船タイタニック号が沈没した日ではないか。彼らはその日のまま閉じ込められているというのだろうか……。

この船には何かが潜んでいる。それも，これだけの空間を特定の時間に閉じ込めておくものが……。船中に潜入し，なんとしてもそいつを殲滅しなければ。

しかし，ここ甲板上にあるはずのタイタニック船中に通じる階段がどこにも見当たらない。なぜ……。タイタニック専門家である私の知識によればここには船中へと通じる階段があるはずなのだ。いや，だいたい甲板に船中へ抜ける道がないなどということがあるのか。何者かが我々の行く手をさえぎろうとしているのだろうか。

＊　＊　＊

さて，彼女らの活躍によってなんとか船のなかに潜入することができた私たちは，

着替え室だ！ デヘデヘ

閃光とともに結界が。キマッたか？

次に私，キディ，香津美の３人と，他のメンバーの２人に別れて事件の鍵をにぎっていると思われる女考古学者と船長を捜すことになった。

さんざん船内を捜しまわり，ついに船長を見つけることに成功したのであるが，船長は私たちのことを不審に思ったようだ。駆けつけた船員に武器を奪われ，南京錠のロックされる音とともに３人は真っ暗な部屋に投げ込まれた。

▲動きのある戦闘画面（でも絵は小さい……）

▶タイタニック号の船内地図

そして，目の前に大きく裂けた口から牙をむき出しにした妖魔が現れた。

「冗談じゃねえぞ，妖魔め」

「キディ，気をつけて」

キディが突進し，妖魔にパンチを叩き込む。バシィ！　しかし，妖魔の爪が振り回され，キディはしこたま地面に叩きつけられてしまう。

「ぐっ！」

圧倒的な妖魔の力。丸腰で対抗するには，もはや香津美の魔法の力しか残されていない。はたして，生き残れるのか……。

由貴ちゃんは俺のもんだ◆◆◆◆◆◆◆

いやいや，書いていて気合いが入ってしまい，思わずサイレントメビウスの語り部になってしまいました。

こういうストーリー展開のこのゲームなのですが，なんとコミック中の主要キャラクター５人(ラリー課長を除いてなんだね)のうち，香津美を除く４人のなかから誰と一緒に船内を回るかを選ぶことができるようになっているのです。サイレントメビウスのファンの性質を考えると，これは非常にいい選択であったと思います。ちなみに私としては由貴ちゃんがやっぱりベストですね。え，なんで記事ではキディと一緒なのかって。うるせえやい，由貴ちゃんは俺のもんだい，ぺぺぺぺぺっ（意味不明）。

PC-9801版では登場しなかったラリー課長のグラフィックも入っているし，ファンの人はひと安心ってとこですかね。

そして，グラフィック。このサイレントメビウスでは，PC-9801版と同じく16色モードを使っているのですが，このグラフィックはなかなかだと思います。16色のタイリングでよくここまで細かいグラデーションを描けたなあとか，よくこんな細かいところまで凝って描いているなあなどと，絵を見ていちいち感動してしまうくらいがんばっています。

特に，細かいところまでよく描き込んでいるという意味ですごいです。普通，ゲームなどで使われるこのテの絵は「あ，グラフィックの容量を削るために髪の毛のわっかを削ったな」とか思える部分がたいていあるものなのですが，このゲームではまったくありません。そうすることで，普通はグラフィックデータが占めるディスクの容量を削ろうとするものなのですが，このサイレントメビウスでは逆にディスクの枚数を多くして，グラフィックを少しでもよくしようとしたのでしょう。なにしろ原作が絵とキャラクターで評判の作品なのですから，原作を生かしたよいゲーム化だといっていいと思います。

また，このゲームはシステム的にも簡単になっています。誰にでもできて，非常に快適です。アドベンチャーゲームにしてはメニューが少なく，LOOK，TALK，MOVEの３つだけだもんね。そのおかげでシナリオもさくさく進んでいきます。戦闘モードでも魔法をかけたり，グラビトンを撃ったりがさくさく。そして，そのときはアニメーション＆効果音もばりばりなので非常に気持ちよいです。

とにもかくにも，絵の緻密さとキャラのかわいさで評判の原作を考えると，その特徴を生かしたいいゲームになったのではないでしょうか。合格点あげていいんじゃないかな，うん。

ただゲームをやっていて思ったんですが，シナリオの詰めというか，考証がちょっと甘いような気がしましたんで，次はどうにかしてほしいな。たぶん，どのキャラクターにしても似たような展開にするためにはこういう話にするしかなかったんでしょうが，「○○(キャラの名前ね)がこんなことするかな？」と思うような行動をとったり，セリフをいったりするのがちょっと気になりました。

え，ゲーム自体が2,3時間で終わってしまうのは短いって？　ううむ，確かにこれを短いと感じる人もいることでしょうが，満足できる内容であればそんなことはとやかくいう必要はないと私は思います。ま，そういう人はまだ一緒に行動してない人と一緒に回ってみれば何回も遊べるんで，得した(?)気分を味わってください。

でも，やっぱり由貴ちゃんでキマリ。

ガイナックスでまる

なんだかんだいっても，出る前は結構心配してました。本当にちゃんと出るのかなと。

特に，キャラの顔にぽつぽつタイリングの跡が見えたりしないだろうなあ，というのは心配してました。なにしろサイレントメビウスは女の子のキャラがほとんどなので，顔がすべすべじゃなかったらかわいそうですもんね（キャラに執着できるこのゲームならではのことなのかもしれないけど）。

結果的には，絵はきれいだし，サイレントの世界をきっちり再現してくれたので別にいうことはない……。あ，いや，ちょっとだけいわせてもらいましょう。それはあのディスクの枚数。きれいなグラフィックをたくさん収めるためとはいえ，さすがにディスク７枚組というのはゲーム中に入れ替えがひんぱんにあることもあって，かなりうざったいです。

７枚のディスクは気分を盛り上げるためだかどうだか知りませんが，それぞれ「封印，胎動，呪縛，彷徨，叫喚，焦燥，鬼哭」などと名づけられています。そして，ディスク入れ替えのときには，“「焦燥」ディスクを入れてください”とかいうメッセージが表示されます。ただでさえ枚数が多くて混乱するのに，これでははっきりいってわかりにくい。せめてもの打開策として，ハードディスクへのインストールがサポートされていれば，よかったと思います。

次回作はいったい何になるんでしょう。私としてはぜひともPC-9801でも評判のアレ，(別名“シムねーちゃん”）をやってみたいんですけど。よろしくお願いいたしますです。

総合評価	0　　5　　10
シナリオ	★★★★★★
効果音	★★★★★★
グラフィック	★★★★★★★
キャラクター	★★★★★★★★★
興奮度	★★★★★★★

愛と勇気と友情と涙？

Yaegaki Nachi
八重垣 那智

発売後，結構時間がたったけど，「パロディウスだ！」のレビューはまだまだ続く。まあ，さすがに今月で打ち止めだけどね。さて，難易度調節があるとはいえ，わりと手強いこのゲーム。君はエンディングを見たか！

すでに説明無用の域に達しつつある「パロディウスだ！」であるが，これはあくまでもゲージパワーアップ方式の横スクロールシューティングゲームである。俗にグラディウスタイプというやつなのだが，通常のメカがわさわさ，カッコイイ戦闘機がバリバリといった趣のあるゲームとは違い，全編ただのギャグになっているのがポイントである。それも「通」の人が思わず誰もいないところで「にやり」とするような，狙ったギャグなのである。ゲームセンターにデビューした当時，いきなりスポーツ新聞に全然似ていないイラストつきで，あやしいページに載ってしまう「パロディウスだ！」は，やっぱりスゴイゲームだったのである。

素晴らしさは「愛」◆◆◆◆◆◆◆◆◆◆

「パロディウスだ！」の売り文句は「完全移植」である。確かにいままでのコナミゲーム（通は「コナゲー」という専門用語を使う）の移植では，ユーザーの満足度を100%引き出せたとはいいがたいものがあった。私のように「ゲームはアーケード！」というタイプにとっては，「クォース」でさえアラが見えてしまったものである。そんなわけで，実はこれも結局「グラフィックコンパチ」のゲームだと，甘く見ていたことは事実だった。しかし，それは出来上がった現物を見たときに，ガラガラと音を立てて崩れてしまったのである。そこにはいままでのものとは違うヤツがいた。私が「なかなかやるわい」などとありもしないアゴヒゲをさすって，動揺を隠したのはいうまでもない。

確かに違うところはある。最たるものは8面のボス，ハリセンボンこと「プ～ヤン」であろう。本来の拡大はもっと滑らかに行われており，ここだけはやはり複雑な気分を感じないではいられなかった。ほかにも先月の中森氏曰くの画面比の問題など，細かい違いは数限りなくあるといえるのだが，全体を見回した結論としていうと，コイツは「完全移植」と呼ぶにふさわしいものであることは，私が保証してもいい。私の保証など何の役にも立たないが，ないよりはあったほうが一家の主も安心というわけである。

あとあまり触れられていないが，音楽の移植もかなり上等である。MIDIを使わずとも，かなりのクオリティを確保しているのは，業務用で鍛えたノウハウのたまものであることは間違いない。確かにPCMのチャンネルが少ないのでそれなりに落ちてはいるのだが，努力でそれをカバーしているので，不平不満をいうのは仁義にもとるというものである。仁義を忘れた悪者は最後に高倉健かなんかにバッサリ切られちゃうので，夜道を歩くときは気をつけたほうがいいだろう。

MIDIについてはあまり具体的に論評できないのだが，いままで聞いたMIDI対応ゲームのなかでは，かなりしっくり受け入れられるものであった。曲調がクラシックのアレンジだということもあるのだろうが，かなりいい。もしMIDIの一式が駅の売店で980円で売っていたとしたら，早速足を運んで780円に値切って買ってくるべきである。私は駅の売店に行ってみたが，どこにも売っていなかったので，MIDIでプレイすることは血の涙を流してあきらめたのだが，持っている人はぜひぜひMIDIを使ってプレイしてほしいものである。

戦うための「勇気」◆◆◆◆◆◆◆◆◆◆

このゲームを買った人は少しがっかりするかもしれない。こいつが結構難しいゲームであることに気づくのには，そんなに時間はかからないはずである。そこでいちおうエンディングを見るためだけのアドバイス程度は書いてみようと思う。攻略というよりエンディングを見ることしか考えに入れてない。ちなみにあなたの身近にうまい人がいて，すでにエンディングを見せてもらっていた場合は役所に行って「エンディング見ました証明書」をもらってくると友達に自慢できるぞ。

まず設定であるが，7機でイージーは常識。エクステンドも下げておいたほうが気休めになるだろう。そうしたらおもむろにゲームスタートだ。その前に精神統一を兼ねて「パロパロパパパ」などと呪文を唱えるのが流行しているらしいので，一度試してみるのがいいだろう。

X68000用　5"2HD2枚組　9,800円(税別)
コナミ　エンタテイメント　☎03(3264)5678

2周目の鬼レーザー，大きさ2倍(当社比)

キャラクターはツインビーで，パワーアップはオート。私は「まず」こいつを選ぶ。本当はマニュアルにしてスピードアップをできるだけ取らないようにして先の面まで行くといいのだが，コンティニューするからランクは下がるので関係ないのである。しかし，下がるといっても難易度のズンドコまで落ちるわけではないので，雨乞いなぞしないように。

2機と2匹の「友情」◆◆◆◆◆◆◆◆◆

で，各面のアドバイスである。

1面　アイランド オブ パイレーツ

猫戦艦アチャコの下の砲台を撃ちにいったときに，飛び降りてきたペンギンに踏まれる凡ミスをしやすい。真似しないように。ツインビーやペン太郎は3回目のオプションゲージが点灯したときに，何かベルパワーを持っていると3つ目のオプションを早く取ることができ，ちょっとお得。

2面　ピエロの涙も3度まで

撃ち負けないようにすること。ミサイル系統の武器をうまく使うといい。ボスは追いかけないで，左端から離れないほうが安全。

3面　ラビリンスお菓子城の謎

ボスでやられた場合は右を3匹だけ倒して時間切れまで粘らないとだめ。片側を全滅させると動き出すので，逆手に取るのがポイント。

4面　嗚呼日本旅情

ボスはカラータイマーを連射。まわし攻撃の前に倒すこと。股間に入って自殺するのも一興。看板には注意。

これは1周目での復活，ここまで粘ろう

地震に耐える屋根裏の住人。本日も落下中？

小豚の正しい耐え方だが，このあとどうする？

5面　宇宙戦艦モアイ

とにかくボス。ひきつけて避けるが基本。弱点は目ということを忘れないように。

6面　軍艦マーチで今日もフィーバー

ここにツインビーのオートパワーアップでのフル装備で来た場合は，カプセルを常にバリアに合わせるようにするだけ。オートなら復活も楽。

7面　ビューティフルギャルズ

敵を見るより自分の周りを見て正確に避けること。この面以降でコンティニューする場合は，ビックバイパーのマニュアルに切り替えて，2速＋ミサイル＋ダブル＋オプション1個の装備を基本にすること。

8面　もっとも北の国から'90

前半がきびしいが，敵の配置を覚えて前で倒す。なかでもイソギンチャクは速攻(ここでビックバイパーに替えて，ミサイルを取れば役立つ)。中盤の弾だらけは，目で見て大胆に避けよう。

9面　ナイト オブ ザ リビングデッド

最後のカラカサが激難。ガイコツの破片より下に入って撃ちきる。途中で赤ベル(ダブルなので調整しやすいはず）を取ってこれたら簡単。菊一文字に重なっていれば無敵であることを利用すること（ほかの場面でもとても有効)。

10面　タコの要塞

グラディウスの最終面のアッパーコンパチ。終盤の上下のハッチは，でっぱってる天井スレスレで地形と一緒に下がりながら撃つこと。

最後の最後でどんでん返し。全治3カ月？

とまあ，ざっと見たわけなのだが，なれてきたら最初にビックバイパーのマニュアルのほうで始めるとコンティニューなしでも可能かと思われる。ルーレットは装備が揃ったら無視してかまわない。回しっぱなしでも害はない。どうしても取るなら，復活ができる場所に来てから押すこと。もしも変なものを取った場合は「ごめんね」とつぶやきながら葬ってあげましょう。化けて出ないから大丈夫。人間思いきりが肝心である。

コナミの「涙」◆◆◆◆◆◆◆◆◆◆◆

「パロディウスだ！」をやっていて感じたのは，X68000のオリジナルの貧弱さである。1年前のアーケードゲームのレベルで傑作級というのは，悲しいことだが現実なのである。でもそれすら出てこないわけではなく，コナミはこんなに楽しいゲームを作ってくれたのである。改めてここでいっておく必要があるかもしれない。「パロディウスだ！」を作ったのは，

大波。コナミ。郷ひろみ～！

というわけで，オリジナルなんかも作ってくれないかなと思う今日この頃である。もしも明日大地震があるなら，コイツをプレイしておいたほうが上から降ってくるガレキを避けやすい，と学会でも提唱されているそうだ。しかし，4面のボスと戦ってるときに地震が起きたら気がつかないから危ないな～，うんうん。

やっぱ移植はクリソツが肝心

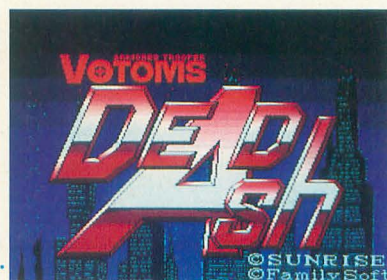
似ているのは写真だけかと思ったら，快くコナミは裏切ってくれました。こんなことならもっとバキバキ移植してほしいですね。縦画面は無理があるから，横画面のシューティングなんかがいいです。「サンダークロス」とか出ませんかね？。まあ，次には「ネメシス」も控えているのでシューティングに不自由する心配はないのだけれども。しかし，本当にここまでできる技術には頭が下がります。ビデオボード使って録画できないっていうのも，なにやら技術の薫りがしてソソりますな。

総合評価	0　　5　　10
ゲーム性	★★★★★★★★★
移植度	★★★★★★★★★
技術力	★★★★★★★★★★
ギャグ度	★★★★★★★★★
いち押し度	★★★★★★★★★★
パッケージ	★★★★★★★★★

THE SOFTOUCH ●装甲騎兵ボトムズ DEAD ASH

遊びじゃないのよ，戦争は

Kaneko Shuniti
金子 俊一

根強いファンを持つテレビアニメ「装甲騎兵ボトムズ」がゲームになった。もちろん，アクションゲームで，画面はコクピットから見た視点を採用。自然と感情移入度も高くなる。ATや武器をうまく選択し，生き残る道を探れ。

私がボンバーマンでノーコンティニュークリア，1500万点プレイヤーの金子です。ほとんどすべてのゲームにあてはまりますが，タイミングが肝心ですよね。

さて，知る人ぞ知るロボットアニメ，装甲騎兵ボトムズがシューティングゲームになりました。ローラーダッシュ・ガシガシ，ミサイル・バシバシのアクションゲームになっています。本当はシューティングなんだけど，アクションといったほうがぴったりくると思います。

うりゃうりゃ，うりゃ◆◆◆◆◆◆◆◆◆◆

基本的に遠くのものは小さく見えるという疑似3Dのゲーム。ステージは全方向スクロール面と強制スクロール面で構成され，全8面となる。全方向スクロール面では一定数の敵を倒すとクリア，強制スクロール面では一定距離を走りきったらクリアになる。

それぞれのステージの最初にアーマード・トルパー（AT）や武器が選択できる。面によって使えるATの組み合わせが違うが，スコープドッグ，ストロングバックス，スタンディングタートルなど，ファンにとってはヨダレものの設定で，ATによって装甲やスピード，回転性能が異なってくる。面に合わせて選択することが重要。武器は銃火器とミサイルポッドをそれぞれ選択。装弾数と破壊力が違う。敵は硬すぎもせず，軟らかすぎもせず，難易度設定はまずまずだ。

スタートそうそう，こんなことをいわれるとは

X68000用 5"2HD版2枚組 8,800円（税別）
ファミリーソフト ☎03(3924)5435

面こりゃこりゃ◆◆◆◆◆◆◆◆◆◆◆◆

それぞれの面のサブタイトルと，私の勧めるAT＆武器選択，攻略法を伝授しよう。EASYモードでも結構難しいぞ。

STAGE0 バトリングで相手を倒せ

全方向スクロール。ストロングバックスでアサルトライフル，ミサイルポッドは以後のステージでも9発を使用。ATの扱い方を徹底的にマスターしよう。レーダーを見ながら，敵を正面に捉える練習をしよう。

STAGE1 敵包囲網を突破しろ

強制スクロール面。ストロングバックス，ペンタトルーパー。最初の難関。戦うべき相手だけと戦って，あとは逃げよう。まともにいくとダメージが大きくなりすぎる。

STAGE2 敵ATを撃ち落とせ

全方向。スタンディングタートル，ソリッドシューター。宇宙面である。強力な兵器を使えば楽勝。弾数が少ないので無駄ダマは撃たないこと。ブライトさんに叱られるぞ（そりゃガンダムか）。

STAGE3 群がる敵を破壊しろ

全方向。ファッティー2，ハンドロケットランチャー。ボスキャラのようなものがいるので，ミサイルはそこまでとっておく。ザコにはロケットランチャーだけで。

STAGE4 ダム・ルークのATを入手しろ

強制。ファッティー2，ハンドロケットランチャー。またも難関。この面に比べるとステージ1はお遊び。いままでとは違った方法でクリアする。

STAGE5 敵基地のすべての兵器を破壊しろ

全方向。ベルゼルガ22型，ハンドロケットランチャー。おちつけばだいじょうV。強制コントロール時は弾が撃てない。

STAGE6 敵戦艦に突入しろ

強制。ブラッドサッカー，ハンディソリッドシューター。がんばるっきゃないね。

武器などの選択はこんなカッコイイ画面で

ゲームおりゃおりゃ◆◆◆◆◆◆◆◆◆◆

どの武器を選択しても自動照準なのはうれしいが，ほぼ正面の敵しか倒せないのは問題。強制スクロール面では，あまり左右に寄れないから正面に捉えるのは大変なのである。

レーダーもついているのはいいのだが，自分のサイズがわかりづらいので使いにくい。障害物が表示されないのも痛い。

結論。もう少し煮詰めると10倍面白いゲームになる可能性大。ボトムズ2が待ち遠しいところ。個人的にはブルーティッシュドッグに乗りたかった。「太陽の牙ダグラム」でもいいけどね。でも，**なんだかんだいって燃えるよ，このゲーム。**

もうちょい，もうちょい

ストーリーがあるのはかまわないが，それを伝えるためにそれぞれのステージの始まりでちょっと待たされるのはイマイチ。また，主人公の会話シーンで，2枚の絵を交互に出すのに毎回ディスクアクセスするのはうっとうしい。メモリが増設してある場合はそちらに読み込んでもらいたい。弾切れの武器は選択できないほうがベター。こんな細かい気配りがほしい。

総合評価 (0〜10)	
音楽	★★★★★
グラフィック	★★★★★★
効果音	★★★★★★★
熱中度（全方向面）	★★★★★★★★★
熱中度（強制面）	★★★★
マニュアル	★★★★★★
キリコになりたい度	★★★★★★★★★

油の切れ目が縁の切れ目

Nishikawa Zenji
西川 善司

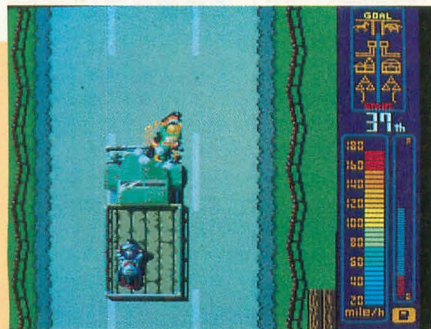
アメリカ各地の市街地ラリーにバイクで挑戦。サンフランシスコ，ロス，フェニックス，デンバー，ボストン，ニューヨーク。そんな設定とはほとんど無関係に道なき道，道ある道をぶっとばせ。それでいいのだ。

私も高校生時代「ダッシュ野郎」と呼ばれていた。4時限目の終業を知らせるベルはまさに校内の「ダッシュ野郎」たちの“GO！”サイン。疾風か？　はたまた「いかずち」か？　教室を飛び出し学食へ走るその姿，これを「ダッシュ野郎」と呼ばずに何と呼ぶ。そう，このゲーム「ダッシュ野郎」も分野は違うが一途で少し不器用な男たちの汗と涙の物語だ。

ダッシュ野郎をやろう

「ダッシュ野郎」はつまりMZ時代からあるBASICの入門本によく載っているサンプルの「昊を左右に動かして上から落ちてくる＊を避けながら♥をとって先へ進め！」ゲームのグラフィックが多少きれいで，BGMがFM音源で自機がバイクになった版と思ってもらってよい。

はっきりいってしまえば，腕まくりをして連射ジョイスティックやらメモ用紙を用意して真剣にプレイしなければならないような大作ゲームではない。ノリはどちらかというとユーザーメイドのショートプロ・ゲームとか学校なんかの電算室にある「タイプ練習ゲーム」のような，単純明快ながらも熱中するとやめられない「かっぱえびせん」的なゲームである。

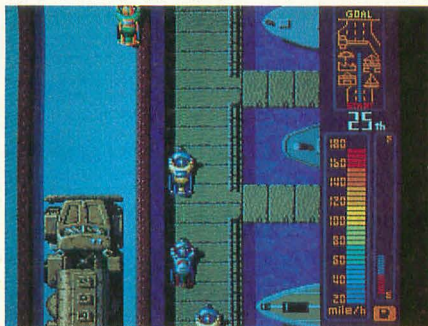
身近にライバルがいれば熱中度は120%（2Pモードもあるし）。つまり「僕，○○点」「へっ，俺なんか××点だもんね」「……くっそぉ」の世界。わかるでしょ。

おらおらおら，トラックのお通りだい

X68000用　5″2HD版2枚組　8,800円（税別）
シャープ　☎03（3260）1161

君はダッシュ野郎だす

さて，ゲームディスクは2枚組，メインメモリが1Mバイトでも動くし，2Mバイトでなくても効果音も鳴る（当たり前）。かっこいい（筆者注：好みによります）ローディング画面のあと，パワー溢れるタイトルが現れる。タイトルに2Pのゲームモードがあるのが目に止まる。が，これは流行の同時プレイのことではなくて，片方のプレイヤーがミスしたら交代するという昔ながらの方式。

最終面以外はコンティニュー回数も無限なので気楽にいこう。あと，なんとこのゲーム，サイバースティックのアナログモードに**完全対応**なのだ。が，そんなものは別にいらない，連射スティックどころかジョイパッドだって不要。キーボードで十二分に遊べちゃうのだ。

ゲーム内容は先にも書いたような単純なもの。じゃあ，「ダッシュ野郎」の魅力ってなに？　それはひとつ，**理不尽極まりない演出**。1面を例に挙げると，ゴール直前で登場するジャンプ台を利用してトラックに飛び乗れちゃって，なんとトラックを操作できちゃう。そしてライバルの乗るバイクをグシャグシャ壊しながらゴールイン，ほとんど犯罪。妖しげなガソリンスタンドや意味不明のボーナスステージもいい味出してる。最終面では**アメ車のデカさの本当の恐ろしさ**を君は思い知るだろう。

こっちを走ったほうが楽

それにしても，このようなヘボイながらもキャッチーな演出の数々。オリジナルを制作した東亜プランってただものじゃないよ，きっと。

「ダッシュ野郎」のもうひとつの魅力はなんといっても**道なき道を走る楽しさ**だ。5面の絶対通れそうにない木々の合間を突っ走ったり，7面の画面のはしっこにある隠れ通路（？）を走ったり……。やりこめばやりこむほど走り方が美しくそして妖しくなっていく……。

そしてダッシュ野郎

もう一度いうが，このゲームは決して大作ゲームではない。しかし，とぼけた演出，間抜けだが憎めないグラフィックなどがあいまって独特の雰囲気を醸し出している。ゲームをえんえんとやっていると視覚，聴覚が麻痺してしまうような気さえする。

「ダッシュ野郎」はきっと夏休み1週間前の学校帰りのような，期待と不安に満ちたCHILDHOODな気持ちを君に思い起こさせるに違いない。フォーエバー「ダッシュ野郎」，君のことは忘れない。

君のハートにガッチンコ

もはや特にいうことはないが，とにもかくにも「ダッシュ野郎」は「ダッシュ野郎」なのだ。真夏の炎天下の朝礼の最中，校長先生の話の途中でブッ倒れてしまったときのような，なんとも遠く懐かしく，それでいてきな臭く……と，書いている本人も形容に困ってしまうようなゲームだ。音楽も安いラジカセで何回もダビングしたようなヘボイ音を出しているが，文句をいわせないパワーと威圧感がある（気のせいという説が有力だが）。とにかく「ダッシュ野郎」，君のことは忘れない。

総合評価	0　5　10
ゲーム性	★★★★★★★
グラフィック	★★★★★
BGM	★★★★★
熱中度	★★★★★★★

魔の8つの湖

Mounai Toshiyuki
毛内 俊行

まだ「遙かなるオーガスタ」にハマっている人も多いと思うけど，オーガスタのコースにはあきちゃったという人もいるでしょう。そういう向きは今回発売された「エイトレイクス　ゴルフクラブ」にぜひぜひ挑戦するべし。

こんにちは。私が自称「オーガスタの魔術師」，毛内です。このたび「遙かなるオーガスタ」の続編「エイトレイクス　ゴルフクラブ」が発売になりました。このエイトレイクスというコース，案外侮れないコースなのです。いったいオーガスタとどこが違うのか？　最初に感じるのは**グリーンがとても速い**ことです。オーガスタと同じ感覚でパターを叩いたりすると，ボールはカップを大きく通りこしてしまうでしょう。慣れないうちはグリーンで3打4打は当たり前かもしれません。

そして，このコースのもうひとつの悩みは**湖がいっぱいある**ということなのです。エイトレイクスというくらいだから，このコースには大小8個の湖が存在します。そして，これらの湖は，我々が打つボールを飲み込もうと，大きな口を開けて待っているのです。ミスショットをしたらボールは湖と思っても間違いありません。

オーガスタでトータル60台前半という輝かしい成績を多数残し優勝経験も豊富な人ほど，エイトレイクスで残すスコアには，大いなる屈辱感を味わうことでしょう。

コースの攻略◆◆◆◆◆◆◆◆◆◆◆◆◆

さて，これほど難しいコースというのは，逆に考えれば攻略するのもなかなか楽しいものです。これからいくつか，面白いホールを紹介しましょう。

池に打ち込めといわんばかり

●Hole3　Par3　175yards

最初に「なんじゃこりゃー！」と思わず叫んでしまうコースです。広い湖の真ん中にぽっかりと浮かぶ島。グリーンはその島の中央に位置しています（実はタイトル画面に描かれているのはこのホールです)。島は幅が狭いので，高度なコントロールが要求されます。島はティーグラウンドよりもかなり低い位置にあるので，クラブは小さめにセットして，バックスピンをかけてワン・オンを狙うといいでしょう。

ティーショットを誤って島の向こう側に落とすと，第3打は湖の対岸から打つことになります。ここからだと距離があるだけでなく，グリーンを隠すように桜の木が植えてあるので，ボールをピンに寄せることよりも，ボールが桜の木にぶつからないようにして島にボールを乗せることを考えます。桜の木にボールが当たった場合，ボールは間違いなく水面に落ちてしまいます。

●Hole9　Par4　465yards

フェアウェイが細く，正確なティーショットを求められるホールです。ティーショットはフェアウェイにあるバンカーの手前です。勇気ある若者は，ドライバーでバンカーの左側を狙いますが，そこは最もフェアウェイが狭くなっていてリスクが大きいのであまり勧められません。スプーンでバンカー手前を狙うのがいいでしょう。

セカンドショットでは，積極的に2オンを狙いましょう。距離的に無理でも，グリーン直前のバンカーからのアプローチなら結構楽にできます。ここで怖いのは，刻んで3打目でグリーンに乗らなかった場合です。ボールは1フィートでもグリーンに近づけ，グリーンを正確に狙える位置をキープすることです。

●Hole16　Par4　440yards

ティーショットは谷越えになります。谷に落ちたらOBになるので，ドライバーでなるべく距離を稼ぎましょう。ティーショットはフェアウェイ左のバンカーの左側を狙うのが楽。ただし，この位置からのセカンドショットは，木がじゃまをして直接グリーンが狙えません。小さめのクラブで一度フェアウェイ中央に出して刻むのが楽ですが，それではバーディは取れません。

断崖に橋がかかっている。やだなあ

バーディを狙う場合は，セカンドショットに大きめのクラブを使い，ドローで大きく立木を巻くようにして2オンを狙います。ただし，失敗すると湖に落ちてしまうので注意が必要でしょう。

戦法練るのもまた楽し◆◆◆◆◆◆◆◆

今日紹介したのはわずか3ホールだけでしたが，エイトレイクスでは18ホールすべてがこんな調子です。つまり気を抜く暇がまったくないのです。しかし，それゆえにバーディを取ったときの感激は忘れられるものではありません。

気をとりなおして頑張りましょう

とにかく難しい。このコースの怖いところは1ホールだけでスコアを3つも4つも崩す可能性があるということ。広いバンカーに捕まったときなど，バンカーから出すのに2打必要なこともあるし，何度も湖へ落とすこともある。でも，それだけやりがいがあるコースだから，スコアが崩れたからといって，すぐにリセットをかけるような真似だけは極力やめてほしい。自分の大きく崩れた成績表を見て得た屈辱感が，明日への大いなるステップとなるのだから……。

総合評価（0〜10）	
難易度	★★★★★★★★★★
コースのセンス	★★★★★★★★
奥の深さ	★★★★★★★★★
熱中度	★★★★★★★
お買い得度	★★★★★★★★

X68000用　5"2HD版2枚組　5,800円(税別)
ティーアンドイーソフト　☎052(773)7770

舞台は己が意のままに

Urakawa Hiroyuki
浦川 博之

面白いゲームに出合って十分に楽しんだあとは，その舞台を自分の好きなように変えたり，あらたに作ってみたくなる。「AⅢ」をプレイしていて，そんな気持ちになったあなたの希望に応えるのが，ズバリこのソフトです。

先月の「A列車で行こうIII」に続いて，今月はそのマップコンストラクションを紹介しよう。その名のとおり，A列車を走らせる舞台となる街を自分の思いどおりに改造するためのソフトだ。

「A列車で行こうIII」をプレイしていると，自分の街やら学校の周りやらをA列車の世界に登場させてみたくなる。特に，僕の場合は家の近くをフニャフニャ蛇行する変な私鉄が走っており，こいつの性根を叩き直してみたかったので，サンプルを渡されるやさっそくマップ作りに取りかかったのだった。

サクッと作ってみよう◆◆◆◆◆◆◆◆◆

操作感覚は「A列車で行こうIII」と同じだ。鉄道や建物の建築はゲームと共通で，メイン画面下のウィンドウが地形作成用に変わっただけである。では，マップ作りのおおまかな流れを説明しよう。

まずマップのどのあたりをどんな地形にするか見当をつけ，LEVEL LANDのコマンドでラフスケッチをする。ペイントツールのBOXFILLの要領で海，市街地，森，畑などを描くのだが，地形ごとに数種類のチップを適当にアレンジして描いてくれるので素人っぽい単調な画面にはならずにすむ。

題材は千葉県某市

次にMOUNTAIN，RIVERのコマンドで山と川を作る。MOUNTAINはポピュラスのように地面を盛り上げ，RIVERはラインを引く要領で描くのだ。家の近所に山はないが，マップに変化をつけるために作ってみた。さらに変化をつけたければ港と空港もある。

細かい修整はTILEコマンドで。家や畑といったチップが並んでいて，これを1コマずつ配置していく。

地形が完成したらDATAのコマンドでマップの名前と最初の予算を決めてしまえばOK。あとは最初から存在する鉄道施設を作る。ゲームにあったコンピュータが動かす鉄道なども もちろん作れる。

コンパチ操作がうれしい◆◆◆◆◆◆◆◆

作ってみると拍子抜けするくらい簡単だった。画面をパッと見ればわかってしまうので，マニュアルもあんまり見る必要がない。思いつくままにいきなり作れてしまうのだ。操作にはあいかわらずアートディンク特有のクセがあるが，「A列車で行こうIII」に慣れてしまった人なら大きな問題ではないだろう。

ニューマップ集，見よこの複雑な路線

ただ，クォータービューの常としてマップの端には半分だけ顔を出しているマスがあるのだが，ここのエディットがしづらかった。何か工夫がほしかったところだ。

それから本物の地図を見ながら描いていて思ったのだが，地形を決める見当がつけづらい。4等分する補助線を入れるとか，作業画面の大きさをマップ全体のキッチリ何分の1かにするとかすれば，もっと作業しやすかったと思う。

この2点を除けば全体に非常に快適で，マップ作成には1時間あまりしかかからなかった。このデータを「マップ登録」で登録し，登録したディスクからユーザーディスクを作成すれば，自分の作ったマップが遊べるようになる。ただ，そのときには作業途中のデータは全部消去しなければいけない。いくつものマップを並行して作っているときにはデータディスクを多めに用意する必要があるだろう。

マップを作ったところまでの評価としては非常に高い点を与えることができる。面倒臭さを感じることはほとんどなかった。作ったマップは他機種と共通で使えるし，「A列車で行こうIII」をプレイした人には迷わず勧められるアイテムだといえる。

X68000用　5"2HD版　3,000円(税込)
新マップ付き　4,000円(税込)
ブラザー工業(TAKERU)☎052(824)2493

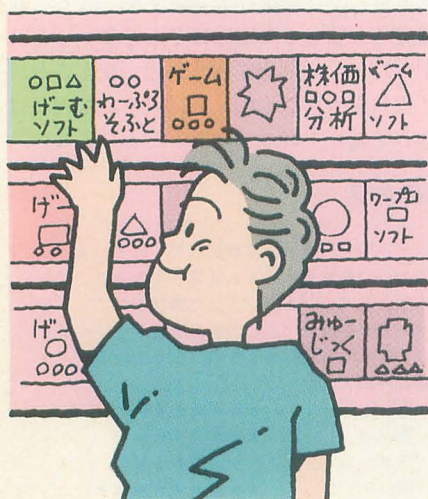
新マップと本のお話

コンストラクションセットに1,000円プラスすると6つの新しいマップが付いてくる。どちらが得かというと断然マップ付きのほうだ。「過密都市を救え」「美しき国立公園」などの正統派マップも楽しめるが，圧巻は「双六鉄道」と「主役は列車」だ。「双六鉄道」は文字どおり，列車をスタートからゴールまで通す双六になっており，「主役は列車」では列車たちが芸術的な団体演技を披露してくれる。これだけでも買う価値はある。

それからA列車のルールの不明確な点についてはビジネスアスキーから出版された「AⅢ THE BOOK」がある程度解明してくれている。効率のいい経営を目指す人は買ってみるといいだろう。各マップをプレイした模様をつづってあるページもあって，このテの本としてはめずらしく読んでも楽しめるものになっている。

総合評価 (0〜10)	
操作性	★★★★★★★
簡単さ	★★★★★★★★
スピード	★★★★★★★
追加マップ	★★★★★★★★★

AFTER REVIEW

海外パソコンで人気を呼び，日本ではPC-9801にいち早く移植されて，これまた人気を呼んだというリアルアクションゲーム(?)がX68000にも登場しました。今月は話題のこのゲームを取り上げてみましょう。

プリンス・オブ・ペルシャ

▶画面と音楽は最高の出来だと思うが，動きがちょっぴし遅いような……。Cコンパイラではなく，アセンブラでプログラムしてほしかった，と思う今日この頃です。ブロダーバンドさん，がんばってくださいね。
谷岡　宙(27)大阪府

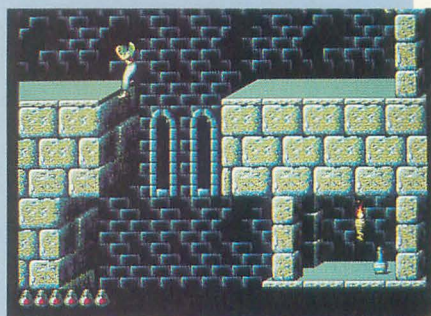
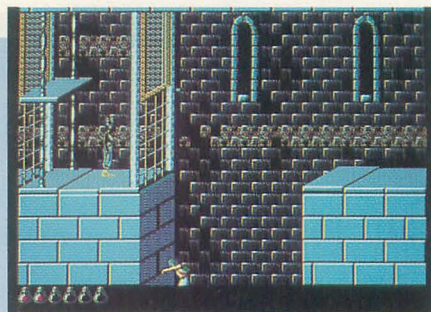
▶「プリンス・オブ・ペルシャ」への危惧は現実になってしまったのだろうか？「カラフルに描き直し」,「デブはよりデブに」といった予告を何かで見たときから心配していたのだが……。IBM PC版などでは色数は少なくても，ドットで打たれた形は厳しい(?)形だったが，これをグラデーションでふくらますと(たとえば，ズボン)ぶよぶよの質感になってしまう。つまり，アウトラインは吟味されたものだとしても，中のふくらみの形が素人仕事になりやすい。アニメーションに見られるように，日本では昔から苦手な部分なんですよね。
北条　章(38)東京都

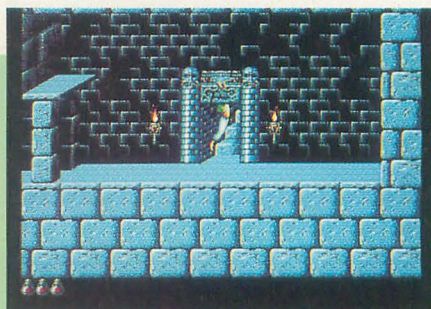
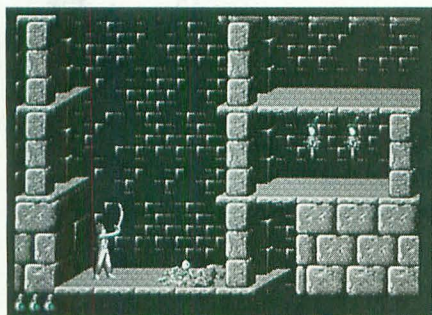
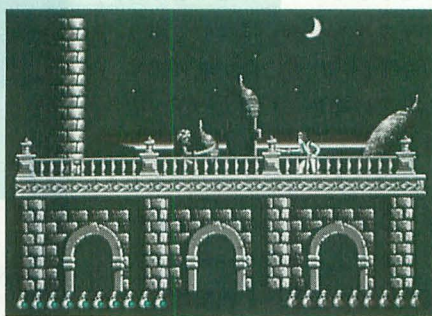
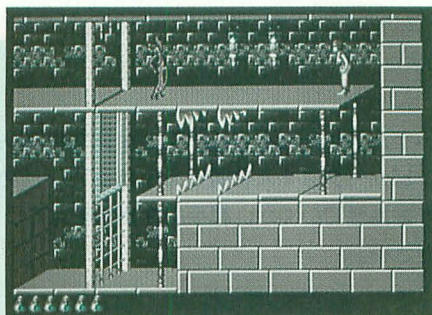
▶「プリンス・オブ・ペルシャ」は買いましたが遅いですね。あんなもんでしょうか。X68000版は機能も縮小だし。まあ，やっているときはまあまあ楽しめますけど（しかし，評判のわりには，ねえ……)。
成海　信之(26)北海道

▶夢の中であるよね。思うように走れないのに逃げつづけようとする自分。あれが体験できる。　小川　知幸(21)宮城県

▶はっきりいって，こりゃひどい。思わずレリクスを思い出してしまうのは私だけではないはずだ。　松本　啓太(19)神奈川県

▶日本のゲームのように，いやらしいトラップがなく，感激するようなトラップが仕掛けられているのがいい。
谷　聡雄(18)茨城県

▶「プリンス・オブ・ペルシャ」に注目し

ていた理由は，もういやというほどいいつくされていることではあるが，その動きのよさにつきる。本物は実際の人の動きを録画したビデオをもとにキャラクターのセルを1枚1枚描き上げていったそうで，素晴らしい動きに仕上がっていた（とある店頭でIBM PC版を見た)。日本版に移植される場合はゲームを見ながら，作り上げていったそうだ。そのため，日本版の場合は動きの滑らかさには少し欠けたものになっていて，違った面白さになっていた。まあ，それはそれでいいと思うが，あくまでも違った面白さ。しかも，それはPC-9801版での話。X68000版はX68000 XVIでやれば，なんとか遊べるというスピードだったので，それ以前の問題という出来であった。
新井　肇(30)神奈川県

▶「プリンス・オブ・ペルシャ」は世界のトップレベルにあるゲームソフトです。

それが，X68000でもプレイできるとしたらユーザーにとって喜ぶべきことでしょう。しかし，移植された瞬間，こんなにもその魅力が損なわれると誰が想像したことでしょう。技術不足によるご苦労は痛いほどわかります。でもX68000に関して適切なアドバイスを受けるだけで多くの問題が解決したはずです。まあ，これが駆け出しのソフトメーカーのオリジナル作品なら「狙いはいいから次回作に期待」で許されるかもしれません。でもプリンス・オブ・ペルシャは一度っきりのビッグタイトルです。次回作はありません。もはやX68000では本当のプリンス・オブ・ペルシャは遊べないのかと思うと悲しくてなりません。ブロダーバンド様，どうか貴社の作品をもっと大切にしてください。お願いします。

志村　一 (28) 東京都

もっと速く！

基本的には，よいゲームだ。反応が悪いと触る気になれないのでファインチューニングしてみた。6Mバイトのを前提にして，私のやったことをすべて書くので，各自，自分の環境にあわせて取捨選択してほしい。

まず，システムディスク。ファイルアクセスが速い可能性があるHuman68k ver.2.0を使い，CONFIG.SYSではBUFFERSをできるだけ広げ，FLOATも新しくする（禁断の/hオプションも加えた)。数値演算プロセッサやIOCS.Xもいいかもしれない。RAMディスクは2.5Mバイト確保し，AUTOEXEC.BATでディスクB，Cの内容を転送する。ファイル数が多いのでRAMディスク内には適当なディレクトリを作り，すべてその中で作業する。VERIFYはOFFだ。

イベントのたびに読み込む音楽データほか，ディスクAのPRINCEの内容はキャッシュに置いた。RAMディスクとキャッシュではキャッシュが速そうなので(あまり根拠はない)。キャッシュへの転送は“COPY ＊.＊ NUL”で行う。私は計測技研のディスクキャッシャを使用しているが，通信などで高速なものを入手していればそちらを使うように。ディスクを抜くとキャッシュがクリアされるのでB，CはRAMディスクに落ち着いた。起動はA:をカレントのまま“C:PRINCE C:”のように行う。

が，これだけやってもほとんど速くならない。次にOPMDRV.Xの高速版OPMDRVX.Xを組み込む（今月あたりの電脳倶楽部に載っているかな？)。これで1割くらい速くなる。同等品が手に入らない場合は音楽をOFFにする。それでも割り込みが消えたか信用できない場合は必殺OPMDRVはずしを行う。あらかじめカレントにOPMという名のファイルを作っておくこと（作れない人は素直にあきらめる)。ゲーム起動後すぐに音楽を止める。マウスには手を触れないこと。

で，XVIで立ち上げる。高速化を換算すれば20MHzノーウエイトのマシンで実行しているのと同等のはずだが……。もともとクロック1MHzの8ビット機のゲームなのに……。

だいたいなんで3枚組なんだ？　絵はいわゆるベロベロ移植だ(PC-9801版は2HD 1枚)。サウンド？　AMIGA版は2DD 1枚でもっと効果音が多くてBGMもサンプリングだ。音の厚みはPC-9801版に劣る。ちゃんとタイミングを取ってないからXVIだと動きがガタつく。重ね合わせ処理も省略されているし，床も現れない。

確かに日本版は動きが粗い。絵を濃やかにすれば動きも濃やかにしないともたないのはアニメの常識。広告の連続写真などを見ると，足の太さがコマによって変わる。さらにズボンに濃い陰がついており，明るい部分の面積が極端に変化しているのでスムースに見えるわけがない（これはアルシスの責任)。

第一人者のA.T.氏の意見では各所のタイミングが海外版に近くなったこと，戦闘でチャンバラできるようになった点がPC-9801版からの改良点だそうだ。XVIなら30分ほどでゲームを終了させるA.T.氏も10MHzではあまり余裕がなかった。初心者には酷なのでは？　（中野　修一）

発売中のソフト

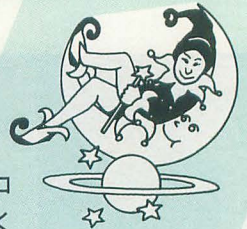
タイトル	メーカー/機種	メディア	価格
ドラゴンウォーズ	スタークラフト		
	X68000用	5″2HD版3枚組	9,800円(税別)
ループス	ブロダーバンドジャパン		
	X68000用	5″2HD版	7,800円(税別)
エイトレイクス　ゴルフクラブ	T&E SOFT		
	X68000用	5″2HD版3枚組	5,800円(税別)
サイレントメビウス	ゼネラルプロダクツ		
	X68000用	5″2HD版7枚組	14,800円(税別)
黄金の羅針盤	リバーヒルソフト		
	X68000用	5″2HD版3枚組	9,800円(税別)
ライヒスリッター	エニックス		
	X68000用	5″2HD版3枚組	8,000円(税別)
装甲騎兵ボトムズ	ファミリーソフト		
	X68000用	5″2HD版2枚組	8,800円(税別)
ダッシュ野郎	シャープ		
	X68000用	5″2HD版2枚組	8,800円(税別)
スターモビール	M.N.M.Software		
	X68000用	5″2HD版	7,200円(税別)

新作情報

タイトル	メーカー/機種	メディア	価格
イース	電波新聞社		
	X68000用	5″2HD版2枚組	9,600円(税別)
生中継68	コナミ		
	X68000用	5″2HD版2枚組	9,800円(税別)
アクアレス	エクザクト		
	X68000用	5″2HD版2枚組	8,700円(税別)
機動戦士ガンダムクラシックオペレーション	ブラザー工業(TAKERU)		
	X68000用	5″2HD版	7,100円(税別)
キャメルトライ	電波新聞社		
	X68000用	5″2HD版	価格未定
フューチャーウォーズ	スタークラフト		
	X68000用	5″2HD版	9,800円(税別)
マジックキャンドル	スタークラフト		
	X68000用	5″2HD版	9,800円(税別)
ダーウィンズジレンマ	スタークラフト		
	X68000用	5″2HD版	9,800円(税別)
eXOn	日本ソフテック		
	X68000用	5″2HD版	価格未定
インペリアルフォース	システムソフト		
	X68000用	5″2HD版	価格未定
ロードス島戦記	ハミングバードソフト		
	X68000用	5″2HD版3枚組	9,800円(税別)
大戦略III'90	システムソフト		
	X68000用	5″2HD版2枚組	9,800円(税別)
銀河英雄伝説 II DX+kit	ボーステック		
	X68000用	5″2HD版	5,000円(税別)
F15ストライクイーグルII	マイクロプローズジャパン		
	X68000用	5″2HD版	価格未定
フェアリーランドストーリー	SPS		
	X68000用	5″2HD版	価格未定
SPINDIZZY II	アルシスソフトウェア		
	X68000用	5″2HD版	価格未定
ドラッケン	エピック・ソニー		
	X68000用	5″2HD版	9,700円(税別)
パワーモンガー	イマジニア		
	X68000用	5″2HD版	12,800円(税別)

X68000CARDDRV用カードゲーム

七並べ

Asai Yasuhiro 浅井 保博

トランプゲームとしてはすっかりお馴染みの「七並べ」です。コンピュータは4人の個性あふれるプレーヤーを担当。地道にいくか意地悪でいくか，相手の性格も考えてうまく勝利をつかんでください。

CARDDRVを使ったトランプゲーム「七並べ」です。

CARDDRVを組み込み，プログラムを実行するとカードが配られ自動的に7を場に出します。このときダイヤの7を出したプレイヤーからゲームスタートになります。自分の番がきたら，マウスカーソルを出したいカードの上に持ってきて左クリックしてください。このとき右クリックするとパスになります。

ルールは誰でも知っているとおり，7からつながるようにカードを出していき，早く手札がなくなった人が勝ちです。出すカードがないときには3回までパスをすることができます。また，出せるカードがあるときでも，ほかのプレイヤーを妨害するためにパスをしてもかまいません。パスは3回までで，4回目にパスをすると破産となり，負けになってしまいます。ほかのプレイヤーをすべて破産させても勝ちとなります。3回のパスをいかに使うかが勝敗のカギでしょう。

（編注：このゲームではカードが両端に達したときのルールがありません。ローカルルールではないと思うのですが，片方の側が端に達すると逆の端からしかカードが置けなくなるというものです。これに限らず，投稿されてくる七並べはすべてこのルールを省略しています。思考ルーチンが大変ですが，七並べを戦略的に面白くするためにはやはり欠かせないと思うのですが）

変数表（グロ一バル変数のみ）

変数	説明
card( )	カード番号をカードが出る順番に格納
ba( )	場に出ているかどうか。出ていないとき0，出ているとき－1
player( )	各プレイヤーのカードがカード番号で入っている
pass( )	各プレイヤーのパスの回数
numc( )	各プレイヤーの持っているカードの枚数
win( )	各プレイヤーの勝ち数
nm( )	各プレイヤーの名前
mx,my,bl,br	マウス用
times	何回ゲームを行っているか
winner	ゲームの勝者
mes( )	メッセージ
shan	カードを出すときの効果音
don	破産のときの効果音
pinpon	あと1枚になったときの効果音
da	パスのときの効果音
agari	ゲーム終了時の効果音

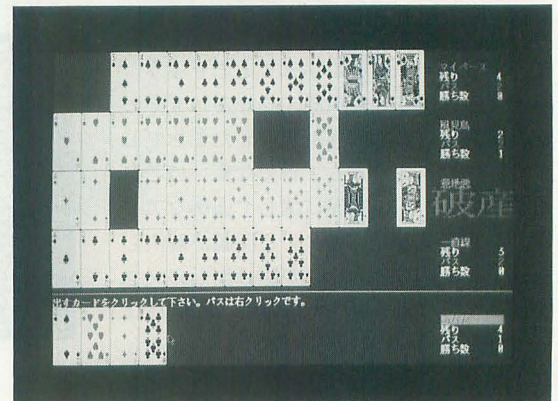
プログラムについて

特に変わったことはしていないので，注釈を見ればだいたいわかるでしょう。シャッフルや効果音などいろいろなところで99を参考にしています。3980，3990行がウェイトになっているので，コンパイルしない場合は削ってもかまわないでしょう。

この「七並べ」では，コンピュータが4人分を受け持つわけですが，この4人のキャラクターはすべて別々の思考ルーチンを使用して，個性をつけてあります。以下にそれぞれのキャラクターの思考ルーチンの特徴を示します。

マイペース

まず自分が上がれるように，それからあまり無理をせずにほかのプレイヤーを妨害します。

風見鳥

ほかのプレイヤーが上がりそうなときなどにパスを使って妨害します。

意地悪

自分が上がることよりも，他人の妨害を優先して行います。

一直線

とにかく自分が上がれるようにプレイします。ほかのプレイヤーを妨害することはありません。

最後に

この七並べというゲームは，プレイヤーの性格がよく表れるゲームだと思います。とにかく早く上がろうとする人，自分が勝つことよりほかのプレイヤーの妨害に喜びを感じる人……。プログラムでもコンピュータの思考ルーチンは，あまり強くありませんが，個性的なメンバーにしてあります。コンピュータの性格を知れば，ゲームはより楽しくなるでしょう。

参考文献

毛内俊行，カードゲームを作ろう，Oh!X 1990年5月号
コンパイラ対応カードゲーム変更点，Oh!X 1991年3月号
その他Oh!X掲載CARD.FNC，CARDDRV用カードゲーム

リスト1

```
  10 /*
  20 /* 七並べ
  30 /*
  40 /*              1991.2.21 - 2.28
  50 /*
  60 dim int card(51),ba(51)
  70 dim int player(4,10),pass(4),numc(4),win(4)
  80 dim str nm(4)=("あなた","マイペース","風見鳥","意地悪","一
直線")
  90 int mx,my,bl,br
 100 int times,winner
 110 dim str mes(4)[95]=("",
 120   "もう一度プレイしますか。左クリックでもう一度、右クリッ
クで終了です。",
 130   "出すカードをクリックして下さい，パスは右クリックです。"
,
 140   "まず7を出します。ダイヤの7を出した人からゲーム開始で
す。",
 150   "まだ出せましたよ。")
 160 mes(0)=space$(95)
 170 str shan="@68o4v1518c",don="@49o2v1514c",pinpon="@54o4v151
16fc8"
 180 str da="@4o2v15132crcrc",agari="@8o4v15116crcf4"
 190 /*
 200 /* main
 210 /*
 220 init1()
 230 repeat
 240   init2()
 250   winner=game()
 260   gameover(winner)
 270 until(replay()=0)
 280 mouse(0):mouse(2)
 290 cls:wipe()
 300 color 3
 310 end
 320 /*
 330 /* 初期設定1（1回のみ）
 340 /*
 350 func init1()
 360 int i
 370 randomize(val(mid$(time$,4,2)+right$(time$,2)))
 380 screen 2,0,1,1
 390 palet(1,0)
 400 console 0,32,0
 410 locate ,,0
 420 color[,rgb(31,0,0)]
 430 m_alloc(1,200):m_assign(1,1)
 440 mouse(4):mouse(1):msarea(0,0,767,511)
 450 times=0
 460 for i=0 to 51
 470   card(i)=i+1
 480 next
 490 for i=0 to 4
 500   win(i)=0
 510 next
 520 endfunc
 530 /*
 540 /* 初期設定2（1ゲーム毎）
 550 /*
 560 func init2()
 570 int i
 580 cls
 590 fill(0,0,1023,1023,8)
 600 fill(0,392,767,393,14)
 610 times=times+1
 620 for i=0 to 4
 630   pass(i)=0:numc(i)=0
 640 next
 650 for i=0 to 51
 660   ba(i)=0
 670 next
 680 sfl(52)
 690 deal()
 700 endfunc
 710 /*
 720 /* シャッフル
 730 /*
 740 func sfl(n)
 750 int a,b,c,i
 760 for i=0 to 99
 770   a=rnd()*n:b=rnd()*n
 780   c=card(a)
 790   card(a)=card(b)
 800   card(b)=c
 810 next
 820 endfunc
 830 /*
 840 /* カードを配る
 850 /*
 860 func deal()
 870 int i,j,k,l
 880 for i=0 to 4
 890   for j=0 to 10
 900     player(i,j)=-1
 910   next
 920 next
 930 i=0
 940 j=rnd()*6
 950 for k=0 to 10
 960   for l=0 to 4
 970     player((j+1) mod 5,k)=card(i)
 980     numc((j+1) mod 5)=numc((j+1) mod 5)+1
 990     i=i+1
1000     if i>51 then break
1010   next
1020 next
1030 sort()
1040 endfunc
1050 /*
1060 /* プレイヤーのカードをソート
1070 /*
1080 func sort()
1090 int c,i,j
1100 for i=0 to 9
1110   for j=i+1 to 10
1120     if ((player(0,i)>player(0,j)) and (player(0,j)>-1)) or
(player(0,i)=-1) then {
1130       c=player(0,i)
1140       player(0,i)=player(0,j)
1150       player(0,j)=c
1160     }
1170   next
1180 next
1190 endfunc
1200 /*
1210 /* ゲーム
1220 /*
1230 func int game()
1240 int i,p,w
1250 for i=0 to 4
1260   prstat(i)
1270 next
1280 pcput()
1290 p=put7()
1300 repeat
1310   if pass(p)<=3 then {
1320     mark(p)
1330     switch p
1340       case 0 : i=man():break
1350       case 1 : i=com1(p):break
1360       case 2 : i=com2(p):break
1370       case 3 : i=com3(p):break
1380       case 4 : i=com4(p):break
1390     endswitch
1400     if i=-1 then {
1410       oto(da)
1420       pass(p)=pass(p)+1
1430       if pass(p)>3 then {
1440         dead(p)
1450       } else {
1460         prstat(p)
1470       }
1480     } else {
1490       cdasu(p,i)
1500       if numc(p)=1 then oto(pinpon)
1510     }
1520   erase(p)
1530   }
1540   w=judge(p,i)
1550   p=p+1
1560   if p>4 then p=0
1570 until w<>-1
1580 return(w)
1590 endfunc
1600 /*
1610 /* ステータス表示
1620 /*
1630 func prstat(n)
1640 int y
1650 y=calcy(n)
1660 color 6:locate 81,y  :print nm(n)
1670 locate 81,y+1:color 7
1680 switch numc(n)
1690   case 0 : break
1700   case 1 : print      " あ と 一 枚 ":break
1710   default: print using "残り        **";numc(n)
1720 endswitch
1730 switch pass(n)
1740   case 2 : color 6:break
1750   case 3 : color 5:break
1760   default: color 7
1770 endswitch
1780         locate 81,y+2:print using "パス       **";pass(n)
1790 color 7:locate 81,y+3:print using "勝ち数     **";win(n)
1800 endfunc
1810 /*
1820 /* プレイヤーのカード表示
1830 /*
1840 func pcput()
1850 int i
1860 for i=0 to 10
1870   switch player(0,i)
1880     case -1 : carddel(i):break
1890     default : c_put(i*48,416,player(0,i))
1900   endswitch
1910 next
1920 endfunc
1930 /*
1940 /* カードを消す
1950 /*
1960 func carddel(x)
1970 fill(x*48,416,x*48+48,511,8)
1980 endfunc
1990 /*
2000 /* 7を出す
2010 /*
2020 func int put7()
2030 int i,j,p
2040 message(3)
2050 for i=0 to 4
2060   for j=0 to 10
2070     if number(player(i,j))=7 then {
2080       if player(i,j)=33 then p=i
2090       cdasu(i,j)
2100     }
2110   next
2120 next
2130 message(0)
2140 return(p)
2150 endfunc
2160 /*
2170 /* カードを出す
2180 /*
2190 func cdasu(i,j)
2200 if i=0 then carddel(j)
2210 cdput(player(i,j))
2220 oto(shan)
2230 player(i,j)=-1
2240 numc(i)=numc(i)-1
2250 prstat(i)
2260 if i=0 then {
2270   sort()
2280   pcput()
2290 }
2300 endfunc
2310 /*
2320 /* 場にカード表示
2330 /*
2340 func cdput(n)
2350 c_put((number(n)-1)*48,(n-1)/13*96,n)
2360 ba(n-1)=-1
2370 endfunc
2380 /*
2390 /* プレイヤールーチン
2400 /*
```

```
2410 func int man()
2420 int c
2430 message(2)
2440 while -1
2450   c=selcard()
2460   if c=-1 then break
2470   if chcard(player(0,c)) then break
2480 endwhile
2490 message(0)
2500 return(c)
2510 endfunc
2520 /*
2530 /* プレイヤーカード選択
2540 /*
2550 func int selcard()
2560 button()
2570 if br then return(-1)
2580 if bl and my>415 and mx/48<11 then return(mx/48)
2590 endfunc
2600 /*
2610 /* 指定のカードか出せるかどうか
2620 /*
2630 func int chcard(c)
2640 int i,j
2650 if c=-1 then return(0)
2660 i=number(c)
2670 if i<7 then j=1 else j=-1
2680 i=i+j:c=c+j
2690 while i<>7
2700  if ba(c-1)=0 then return(0)
2710  i=i+j:c=c+j
2720 endwhile
2730 return(-1)
2740 endfunc
2750 /*
2760 /* 終了判定
2770 /*
2780 func int judge(p,i)
2790 int j=-1
2800 if numc(p)=0 then return(p)
2810 if i<>-1 then return(-1)
2820 for p=0 to 4
2830   if pass(p)<=3 then {
2840     if j=-1 then j=p else return(-1)
2850   }
2860 next
2870 return(j)
2880 endfunc
2890 /*
2900 /* 破産
2910 /*
2920 func dead(p)
2930 int y,i,j,k
2940 for i=0 to 10
2950  for j=0 to 3
2960    for k=0 to 1
2970      if chcard(player(p,i)) then message(4)
2980    next
2990  next
3000 next
3010 y=calcy(p)+1
3020 for i=0 to 2
3030   locate 81,y+i:print space$(13)
3040 next
3050 oto(don)
3060 symbol(632,y*16,"破産",3,2,2,5,0)
3070 allput(p)
3080 endfunc
3090 /*
3100 /* 破産した時、すべてのカードを出す
3110 /*
3120 func allput(p)
3130 int i
3140 for i=0 to 10
3150   if player(p,i)<>-1 then {
3160     cdput(player(p,i))
3170     if p=0 then carddel(i)
3180   }
3190 next
3200 endfunc
3210 /*
3220 /* 1ゲーム終了
3230 /*
3240 func gameover(p)
3250 int y,i
3260 oto(agari)
3270 y=calcy(p)+1
3280 for i=0 to 2
3290   locate 81,y+i:print space$(13)
3300 next
3310 symbol(632,y*16,"勝利",3,2,2,3,0)
3320 win(p)=win(p)+1
3330 color 7
3340 for i=0 to 4
3350   locate 81,calcy(i):print using " ##戦 ##勝";times,win(i)
3360 next
3370 endfunc
3380 /*
3390 /* マウスのボタンが押されるまでウエイト
3400 /*
3410 func button()
3420 repeat
3430   msstat(mx,my,bl,br)
3440 until bl or br
3450 mspos(mx,my)
3460 endfunc
3470 /*
3480 /* もう一度プレイするか
3490 /*
3500 func int replay()
3510 message(1)
3520 button()
3530 message(0)
3540 if bl then return(-1)
3550 return(0)
3560 endfunc
3570 /*
3580 /* メッセージ表示
3590 /*
3600 func message(i)
3610 locate 0,25:color 7
3620 print mes(i)
3630 endfunc
3640 /*
3650 /* 該当プレイヤーを明示
3660 /*
3670 func mark(p)
3680 int y
3690 y=calcy(p)*16
3700 fill(648,y,752,y+15,3)
3710 endfunc
3720 /*
3730 /* マークを消す
3740 /*
3750 func erase(p)
3760 int y
3770 y=calcy(p)*16
3780 fill(648,y,752,y+15,8)
3790 endfunc
3800 /*
3810 /* Y座標の計算
3820 /*
3830 func int calcy(p)
3840 int y
3850 switch p
3860   case 0 : y=27:break
3870   default: y=(p-1)*6+1
3880 endswitch
3890 return(y)
3900 endfunc
3910 /*
3920 /* 効果音
3930 /*
3940 func oto(m;str)
3950 m_init()
3960 m_trk(1,m)
3970 m_play()
3980 while m_stat(1)
3990 endwhile
4000 endfunc
4010 /*
4020 /* 思考ルーチン1（他のカードが出せるように出す）
4030 /*
4040 func int think1(p)
4050 int i,j,k=-1
4060 for i=0 to 10
4070   if chcard(player(p,i)) then {
4080     k=i
4090     for j=0 to 10
4100       if i<>j and (player(p,i)-1)/13=(player(p,j)-1)/13 an
d sgn(number(player(p,i))-7)=sgn(number(player(p,j))-7) then ret
urn(i)
4110     next
4120   }
4130 next
4140 return(k)
4150 endfunc
4160 /*
4170 /* 思考ルーチン2（7から遠いカードから出す）
4180 /*
4190 func int think2(p)
4200 int i,j=-1
4210 for i=0 to 10
4220   if chcard(player(p,i)) then {
4230     if j=-1 then {
4240       j=i
4250     } else {
4260       if abs(number(player(p,i))-7)<abs(number(player(p,j)
)-7) then j=i
4270     }
4280   }
4290 next
4300 return(j)
4310 endfunc
4320 /*
4330 /* 思考ルーチン3（1+2）
4340 /*
4350 func int think3(p)
4360 int i
4370 i=think1(p)
4380 if i<>-1 then return(i)
4390 return(think2(p))
4400 endfunc
4410 /*
4420 /* カード番号からカードの数を得る
4430 /*
4440 func int number(i)
4450 return(((i-1) mod 13)+1)
4460 endfunc
4470 /*
4480 /* コンピュータルーチン1
4490 /*
4500 func int com1(p)
4510 int i
4520 i=think1(p)
4530 if i<>-1 then return(i)
4540 i=think2(p)
4550 if i<>-1 then if pass(p)<2 and numc(p)>1 and abs(number(pl
ayer(p,i))-7)<3 then return(-1)
4560 return(i)
4570 endfunc
4580 /*
4590 /* コンピュータルーチン2
4600 /*
4610 func int com2(p)
4620 int i
4630 for i=0 to 4
4640   if numc(i)=1 and i<>p and pass(p)<3 then return(-1)
4650 next
4660 for i=0 to 4
4670   if pass(i)<=pass(p) and i<>p then return(think3(p))
4680 next
4690 return(-1)
4700 endfunc
4710 /*
4720 /* コンピュータルーチン3
4730 /*
4740 func int com3(p)
4750 int i
4760 for i=0 to 10
4770   if chcard(player(p,i)) and abs(number(player(p,i))-7)>3
then return (i)
4780 next
4790 if pass(p)<3 and numc(p)>1 then return(-1)
4800 return(think3(p))
4810 endfunc
4820 /*
4830 /* コンピュータルーチン4
4840 /*
4850 func int com4(p)
4860 return(think3(p))
4870 endfunc
```

画像処理と称して遊ぶ

絵が描けない大人でも，絵を加工，編集したりはしたい。ということで，荻窪氏はイメージユニットを使って，スチルビデオカメラやただのビデオカメラなどから画像を取り込んだり，それを加工したりということをはじめたようです。今回はそのお話。

Ogikubo Kei 荻窪 圭

例によって例のごとく，Multiwordは間に合わなかった。よって，いままでとは全然関係なくて，特集とも関係ありそうでないという『夏休み自由研究』といってみたい。大人にだって（最近のサラリーマンなら）夏休みの10日くらいあるのである。

というわけで，今回はたった8行の前書きだけで本題に突入する。夏休みだからである。

1:必要なもの

・道具
X68000（要2MB，できたら4MB）本体
ハードディスク（あったほうがいい）
ディスプレイ（テレビ内蔵だとなおよい）
カラーイメージユニット
電子スチルカメラ
・調味料
Z'sSTAFF PRO-68K ver.2.0
Zs－EX（Oh!X '91年1月号付録にあり）
・素材
適当な被写体

2:何をなさんとするか

X68000はハイなグラフィック能力を持っている。わざわざいわんでもいいことである。

じゃあ，それで何をするか。何はさておき，ソフトの質に影響を与える。ファランクスはX68000だからあそこまでできたのである。ちなみに，ファランクスをEASYモードでコンティニューを2回くらいやって，やっとこさクリアしたぞ。ああ，疲れた。シューティングにここまで燃えたのはグラディウス以来だ。満足満足。

で，これではなんのために前書きを8行ですませたのかわからないので，さっさと次へいく。

マウスで絵を描くのは大変である。とっても大変である。特に，65536色なんて使いこなそうと思ったら，とてつもない労力がいる。我々にそんな画才はない。

じゃあ，どうするか。絵なんて適当にそのへんから持ってくればいいのである。そのためのカラーイメージユニットであり，スキャナなのである。それでいいのだ。

今回はそのカラーイメージユニットで遊ぶ。カラーイメージユニットってのはそれだけでは何もしてくれない。ビデオ信号を入れてやってはじめて仕事をする。ソースが必要なのだ。ビデオ信号を出力する機器といえば，ビデオデッキかビデオカメラ，そして，スチルビデオカメラなどだ。

私がスチルビデオカメラに興味を持ったのは，某ネットワークのオフラインミーティング（パソコン通信仲間の飲み会）ってやつに出席したときだ。そのなかのひとりが，キヤノンのQPICを持ってきていた。どうやら，パソ通愛好者にQPICユーザーってのは多いらしい。マビカではなく，QPICってとこがミソ。どういうわけか，みんな，QPICなのだ。その画像をX68000に取り込んで，アップロードしたりもするらしい。

そんなこんなで，そのQPICユーザーはX68000ユーザーだけどカラーイメージユニットを持っていないというので，我が家でX68000に取り込んでみたわけだ。

あまりきれいではない。色の再現性もいまいちだし，取り込んだときにノイズやモアレが生じるため，鮮明さに欠ける。家庭用ビデオカメラで撮った絵よりはまし，ってなところだ。

だが，ちょっとぼかしをかければノイズもとれるし，そこそこ見られるものになる。CGとはひと味違った，自然画独特のくすみ，みたいなものもまた楽しい。

何よりも魅力なのが，スチルビデオカメラの機動力だ。一般の写真のように現像の手間もいらないし，ランニングコストも安い。2インチのフロッピーディスクも，うまく撮れたものだけ，X68000に落として管理すれば，なかなか手軽なアルバムが出来上がる。

確かに，スキャナのほうがカラーイメージユニット＋スチルビデオカメラのセットよりも安いし，クオリティも高い。スキャナで取り込んだグラフィックとカラーイメージユニットで取り込んだグラフィックを比べれば月とスッポンだ。それでもやはり，どんなものでも撮れてしまうスチルビデオカメラの機動力に魅力を感じるのである。

今回行うのは，スチルビデオカメラによる画像入力。それから，それだけではつまらないので，Z'sSTAFF PRO-68KとZs－EXを使って行う画像合成だ。

3:スチルビデオカメラとは

第0期のスチルビデオカメラは試作されたマビカだったが，第1期のスチルビデオカメラはQPICが代表だ。名前のとおり，静止画専用のビデオカメラだと思えばいい。受光部はCCD。

CCDが得た画像（光）は，ビデオカメラと同じく，磁性面に記録される。テープではなく，ディスク。ソニーの作った，2インチのフロッピーディスク。プロデュースが採用しているのと同じディスクだが，パソコンやワープロのメディアとしての2インチフロッピーディスクと，スチルビデオカメラの記録用2インチフロッピーディスクは，同じものではあるけれども，使い方に決定的な違いがある。

コンピュータのメディアとしての2インチフロッピーディスクはデジタル記録だが

(当たり前だ)，ビデオフロッピーディスクはアナログ記録なのである。考えてみたら，あんな小さいディスクにフルカラーの画像をデジタルで記録したらどれだけの容量が必要となるかはX68000のグラフィック用RAMが512Kバイトあることを思えば推して知るべし。それが，2インチのディスクに50枚もの画像が記録できるのである。画質がいいわけがない。

さて，QPICをはじめとする(マビカとかもあるが)第1世代では，コストダウンのためか，フォーカシングは“写るンです”と同じ固定焦点。“写るンです”と同じでズームもない。ストロボと逆光用ボタンがついている程度だった。

今回私が試用したスチルビデオカメラは第2期にあたるもの。ズーム，オートフォーカス，マクロという必須の3機能がついたものだ。やっと人並みね，って感じだ。セルフタイマーもついている。しかも，毎秒5ないしは15コマの連続撮影ができる。毎秒15コマだ。こんだけできれば十分動画だが，1枚のディスクを3.33秒で使い切ってしまうので注意だ。

そいつは，オリンパスのスチルビデオカメラ「VC-100」。誰も知らないだろうと思う。なぜなら，店頭販売をしていないからだ。そういうものを私が買ったのかというとさにあらず，オリンパス光学のご好意により，借りたものである。だから，宣伝でもしてあげようと思わないでもない。

「VC-100」は前述したような機能を持っている。補足すると，CCDは1/2インチで，シャッタースピードは1/30～1/2000秒。感度はISO 160相当。レンズは35mmフィルム換算で，54～147mmだ。ストロボは自動発光と発光禁止の2つのモード。ホワイトバランスはオート。こんなとこかな。

あとは写真1を見ておくれ。

両方で194,600円。高いね。QPICなんて，いまや実売価格は6万円以下だから。高いのは，新製品だから，というより，スチルビデオプロセッサっていう箱がついてくるからだ。こいつは何をするかというと，リモコンを使って，ズーミングをしたりランダムアクセスをしたりする装置。このスチルビデオプロセッサとアクセサリ(ACアダプタやバッテリなど)とカメラが基本セット。ほかにプレゼンテーション用として，シャープの液晶テレビとケースが加わったビジュアルシステムもある。さらに上等なセットだと，スチルビデオレコーダってやつが加わる。高級な録画/再生デッキだと思えばいい。RS-232Cでの制御もできる。このへんになってくるとかなりのお値段なので，まあ，気にすることもないだろう。

あと，カラーイメージユニットで画像を取り込むには関係ないけど，S端子を持っているから，S端子つきのモニタにつなぐと，もっときれいなはずである。

4:スチルビデオカメラとカラーイメージユニットの整合性

さて，スチルビデオカメラのビデオ出力端子とカラーイメージユニットのビデオ入力端子をつなぐのだが，ここでひとつ問題がある。512×512ドットのモードで使うとき，X68000はインタレースするのである。入力画像も，インタレースでないと受けつけない。

インタレースってのは，524本ある走査線の奇数番目と偶数番目を交互に出力する方式。テレビジョンがそうしているから，通常のビデオデッキの信号もそうだ。ビデオデッキによっては，静止画のときだけフレームバッファに絵を落として，インタレースで出しているものもある。静止画像時のノイズをなくすためだと思う。

どうしてインタレースなのかというと，当時のテレビ技術のせいだ。間引きしてごまかしているのだな。X1なんかはインタレースしてないから，なんとなく間引きされたようなスーパーインポーズしている。だから，スーパーインポーズは640×200ドットのときに限られたわけだ。400ドットだと，インタレースしないと全部を出力できないからだな。X68000の場合，512ドットモードならインタレース，256ドットモードはノンインタレースだ。

で，X68000でスーパーインポーズすると，オーバースキャン（画面にパソコンの画面が納まらず，はみでてしまうこと）するのだが，これは，524本のうち，画面に現れるのは500本以下だからだ。

本題に戻る。スチルビデオカメラにはノンインタレース信号しか出さないものがあるのである。ノンインタレース信号しか出さないやつが相手だと，カラーイメージユニットで画像を取り込むことができない。荻窪主調べによると，サムライフロッピーはノンインタレース信号らしい。マビカはインタレース信号らしい。オリンパスの「VC-100」はインタレース信号だ。問題はQPIC。隠しスイッチがあって，そいつを先のとがったもので切り替えると，インタレースとノンインタレースの両方を使える。隠しスイッチで切り替えるってのがミソだね。デフォルトの状態ではノンインタレース信号のようだ。

スチルビデオカメラをX68000への画像入力機器として使うことを考えた場合，チェックするのはここだけである。

ちなみに，MacintoshやNeXTの世界(つまり，キヤノンさんですが）では，スチルビデオカメラを画像入力機器として使用し

❶本体の「VC-100」とスチルビデオプロセッサ「VA-200」

ようという試みがある。キヤノンがSCSIデバイスとして，２インチのビデオフロッピードライブを発売しているのだ。画質はどんなもんだか，っていうと，それほどのもんじゃない。29万円という価格が高いというのは確かだ。SCSIだってことは，仕様さえわかればX68000でも使えるということだ。もう20万円ほど安ければね。残念。

5:都庁を撮る

さて，スチルビデオカメラで撮った画像をカラーイメージユニットを通してX68000に読み込む。せっかくカメラ側にＳ端子が出ているのだから，Ｓ端子に対応したカラーイメージユニットがほしいよね。

写真２は私の名刺を撮ってみたところ。まあ，マクロを使えば，こんなもんだ。なんとか読める。なお，取り込んだ画像はフルスクリーンではない。これは512ライン分も画像がないからである。住所と電話番号が目隠しされているが，それは取り込んでからやったものだ。

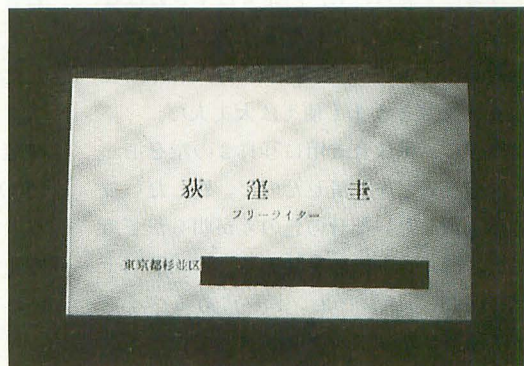
写真３は新都庁。わざわざ新宿まで行って撮ってきたものだ。Z'sSTAFF PRO-68Kで取り込んでいる。Z'sSTAFF PRO-68KはコントラストやカラーTINTなどをマウスで調節できるので，ほどよいところを選ぼう。元の画像（ビデオをそのままモニタするのと比べると，どうしても色の鮮やかさは再現できない）と同じというわけにはいかないが，好みのところで取り込む。大逆光の状態で撮ったので，無理やりコントラストを上げて取り込んだら，こうなってしまった。元の絵は真っ黒だったのだ。写真４は，都庁のツインタワーとその前にくっついている“都議会議事堂”である。渋めに，コントラストを落として取り込み，コメントを入れてみた。なんと，ツインタワーだけが都庁ではないのだ。すごいね。はは。だいたい，新宿の真ん中に見合うデザインじゃあない。海の真ん中（写真５）に建てるとか，立川基地跡にするとかすればよかったのに。

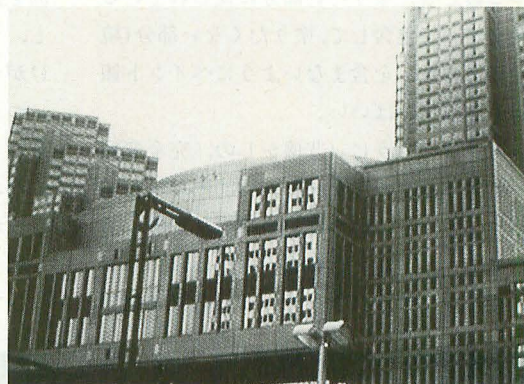
新都庁はどうでもいいとして，こうやって撮りだめしたのはX68000に取り込んでPICファイルでセーブする。写真３を何の加工もせずにそのままセーブしたら200Kバイトになった。ベタだったら512Kバイトなのだから，ここまで圧縮してくれるPICってすごい。

で，たいていの場合，磁性面節約の意味からも，トリミングして保管する。それから，X68000に落としたメリットとして，空いた場所にコメントを入れられる。さらに，写真より管理が楽だ。ディレクトリをうまく使って，ファイル名をうまくつければ，検索だって簡単にできるようになる。

うーん。楽して画像データベースだな。マルチメディアだな。APIC.FNCを使えば，BASICで画像データベースの出来上がりだ。

さて，写真をよく見てもらえばわかるが，ノイズが入ったりビデオ画像取り込み特有のノイズが出たりしてあまりうまくない。もともとディテールは期待していない，ということで，全体にぼかしをかけてしまおう。ぼかしはMFGEDっていうフリーウェアのぼかし機能がなかなか自然画像をきれいにしてくれる。Z'sSTAFF PRO-68Kだと，ぼかし具合をいろいろ試す必要がある。あまりソフトフォーカスしたくない人は，適度なところを見つけよう。写真６が写真３をちょいとトリミング（っていっても，いらないところを黒でボックスフィルしただけだけど）し，ぼかしをかけたところだ。

ぼかしのメリットはもうひとつある。ぼかしによって隣接したドットの色が同じ，というケースが増えるので，PICしたとき

❷とりあえず私の名刺を取り込んでみた

❸そそり立つ都庁

❹都庁のオマケ，都議会議事堂

❺合成で海の真ん中に都庁を

❻写真３をトリミング＆ぼかし

の圧縮率がよくなるのだ。写真6をPICした結果，約125Kバイトにまでなった。このくらいなら，1枚のディスクにけっこう大量に突っ込める。

簡単に，Z'sSTAFF PRO-68Kで加工してPICでセーブってなことを書いたが，もちろん，そのときはZs—EXかPICFILERを使うのだ。

6:都庁で遊ぶ

さて，Zs−EXで立ち上げたZ'sSTAFF PRO-68Kで画像を読み込むのだが，こいつは実にメモリを食う。要2Mバイトどころか，Z'sSTAFF PRO-68Kの機能をちゃんと使おうと思ったら，FEPもはずさねばならないくらいだ。それでも，アンドゥはできない。アンドゥはZs−EXのAltanative Screen機能をうまく使えば大丈夫だ。

つぎに，画像合成用に写真3の空を消した。空を青で塗り潰したのだ。残したい被写体と消したい部分の色の差が明らかであれば，ペイント機能を使える。Z'sSTAFF PRO-68Kのペンウィンドウにあるペイントアイコンをダブルクリックして，ペイント範囲を指定するのだ。

このとき，塗りたい部分に使われている色をすべて包含して，塗りたくない部分(境界線あたり)を含まないようにペイント領域を調節すればいい。

写真7のように(昔懐かしの，完全変形バトロイドバルキリーだ。3,980円だったと思う。しかもこいつは，当時の私が暇にあかせてぺたぺたと色を塗ったうえに，長年の埃とヤニで汚れきっている)，背景と物体の境界がはっきりさせられないときは，ペイントする色(青)でその物体の周りを囲み，それからその境界線のみを含まない色をペイント領域に指定して，塗る。と，写真8になる。

さあ，バルキリーを都庁の前に立たせてみよう。

7:都庁とバルキリーの合成

都庁とバルキリーを合成してみる。

合成にはZs−EXを使う。Zs−EXは2つの画面を持てるので(Altanative Screen機能)，その両方に都庁の画面と写真8を取り込んでおく。そして，MAP機能を使って重ね合わせをするわけだ。

方法は2つ。ひとつは，都庁だけマスクした画面に，写真8を重ねる。都庁だけマスクするには，Zs−EXのMASK PAINT機能を使って空をマスクし，Z'sSTAFF PRO-68K本体に戻って，マスク反転をする。そして，またZs−EXにいって，MAP機能を使って位置を合わせて貼り込めばいいわけだ。

もうひとつは，写真8の背景をマスクペイントし，Altanative Screenして都庁の画面を前面に出し，MAP機能で重ねる。すると，マスクされていないバルキリー本体だけが合成される。

つまり，マスクを使って抽出したほうを重ねるか，マスクして保護したほうに重ねるかの違いだ。今回はどちらを使ってもできるが，空を抜いていない都庁に重ねようと思ったら，後者の方法しかない。

さて，合成写真が出来上がった。なんならバルキリーを少しずつずらして並べてもいい。ちょっとずつ拡大しながら，ってのも計算時間がかなりかかるが，可能だ。

さて，背景だ。とりあえず，空。が，間抜けな私は，空だけのシーンを撮り忘れていたのだ。写真3の空をそのまま使ってもいいが，あまり色がよくない。

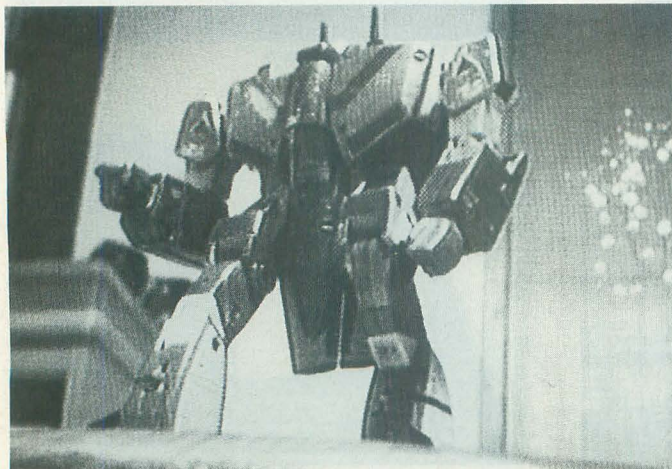
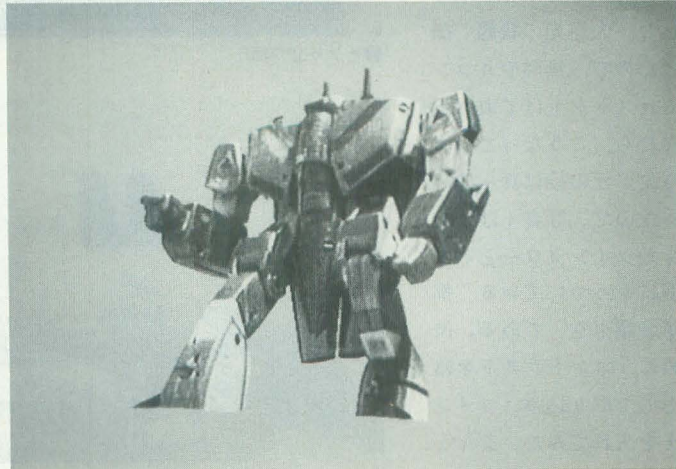
そこで，応急処置として，Macintoshの1600万色フルカラーモードで，Photo Shopを使って空を描き(といっても，それっぽくグラデをかけただけだが)，スチルビデオカメラで写して，X68000に取り込んだ。

こいつを背景に入れ，最後の仕上げに，ぼかしをかける。完成品が写真9だ。

だいたい，こんなもんである。簡単なお遊びしかしなかったが，Zs−EXを使えば，ビルの壁に貼りついたジェニーちゃん(写真10。いっておくが，私の人形じゃないぞ。偶然，この人形をおもちゃ屋で買ってからうちに遊びにきた友達がいたのだ。このジェニーはJALのスチュワーデスバージョンらしい。上着と鞄は外してある)とか(3D写像を使って角度を合わせる)，幅の広い都庁とか，富士山麓の都庁とか，いろいろと遊ぶことが可能だ。

8:サンプリングの時代

スチルビデオカメラは現像処理がいらない，機械を通さないと画像が見られない，というプライベート色の強い画像処理アイテムだから，誌面にはお見せできないようなエッチな写真も安心して撮れる。

画像のハイクオリティを求めない分野では，仕事にも使える。

❼バルキリーを取り込んで……

❽背景を消す

先月号の『大人のためのX68000』でCARD PRO-68Kの背景に使われている絵は，X68000を写して，それを背景にしたX68000を写して，それを背景にしたX68000を写して……，ということをやってみたものだ。こんなこともできる。

それに，スチルビデオカメラはディテールに凝れない代わりに，スキャナなんかの画像に比べて，アバウトにいじることが可能だ。これもうれしい。

X68000は65536色の画像を扱うことができる。しかし，普通の大人にはそれをマウスで描ききることは不可能だ。じゃあ，自然画像を扱おう。どうせなら，少しくらい質は落ちても，自分で捉えた画像をリアルタイムで見たい。じゃあ，スチルビデオカメラはどうだ。てなわけだ。

別にスチルビデオカメラでなくても，ビデオムービーでもいい。それどころか，テレビや市販のビデオから画像を取り込んでもいい。記事にするときには，著作権という面倒なものがあるから，今回は避けたのだが，ソースは何だっていいのである。

いっさいがっさい取り込んで，グチャっとやって，コラージュしてもいい。ハウスミュージックなんて，サンプリング音ばしばしで，それも過去の他人のレコードからサンプリングしてたりする。

それがデジタルの時代というものである。デジタルの時代，データは際限なくコピーされるものなのだ。デジタルデータに移動という概念はない。

だから，どんどん画像を切り取って，どんどんつなぎ合わせて，自分なりの何かを作っていけばいい。

自分で撮った画像を集めて，画像データベースっていう実務的な使い方より，個人的には，大コラージュ大会が好きである。合言葉は，“デジタル化されたデータに節操はない”。そういうものである。

9.カラーイメージユニットIIとPhotoShop PRO-68Kとは

んなものはない。が，カラーイメージユニットが出てから4年になる。もうこれでは許されない。質的にも価格的にも配線の面倒臭さ的にも許されない。

まず，うっとうしい配線をなんとかすること。ついでに，S端子に対応すること。クオリティをアップすること。ノンインタレースでも読み込めるようにすること。それでもって，価格は39,800円くらいだな。個人的には，X68000に内蔵してもいいと思っている。

しかし，それだけでは片手落ちで，カラーイメージユニットIIに合ったアプリケーションが必要だ。仮称として，PhotoShop PRO-68K という名にしよう。PhotoShopっていうのはMacintosh用のフォトレタッチ用ソフトだ。日本語版で，15万円もする。

フォトレタッチっていうのは，つまり，写真を修整したり加工したりすることだ。このソフトにはそのための多くの機能が盛り込まれている。

X68000にもこういうのが必要だ。Z'sSTAFF PRO-68Kはしこしことドット修整する道具ではあるが，アバウトに，画像をザっと加工したりエフェクトしたりするには貧弱すぎる。カラーイメージユニットIIで取り込んだ画像をいじるには，専用のそのテのアプリケーションが必要だ。

せっかく映像屋さんが作ったパソコンなのだから，こういう方面もしっかりおさえなければいやである。

10:追伸と予告

と，ここまで書いたところで，新しいアイテムを衝動買いしてしまった。

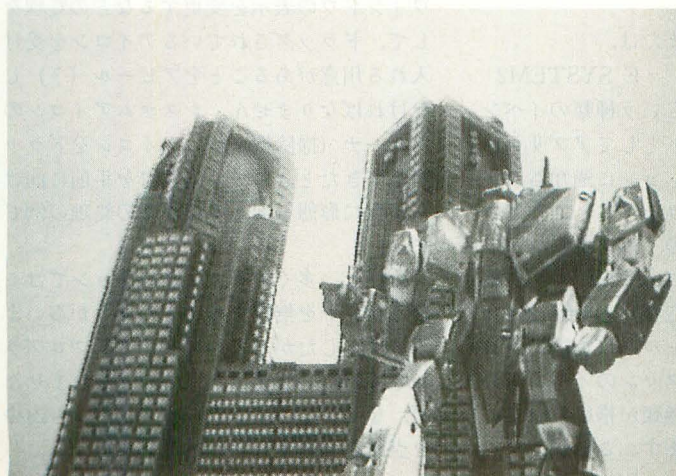
ふふふ。待ちに待った，Hi8のハンディカムである。私が買ったのはソニーのOEMである京セラの製品なのだが，中身はおんなじである。

これがまた，なかなかきれいな絵を撮ってくれるのだ。Hi8を待ってよかった，って感じ。前のハンディカムには画質の点で不満があったからね。

なんといってもCCDは41万画素である。今回使用したオリンパスのスチルビデオカメラが36万画素，ブレンビーが25万画素だそうだから，やっぱ，いいのである。カラーイメージユニットで取り込んで見たところ，発色が鮮やかでなかなかのものであった。

世間の映像機器がいまのテレビ規格の限界に向かって進歩しようとしているとき，カラーイメージユニットがあれではね，って思う。

さて，来月の『大人のためのX68000』は，今度こそ，「Multiword PRO-68K」である。鬼が出るか蛇が出るか，お楽しみである。Macintoshを買ってDTPの世界に触れてしまった荻窪圭はちょっとうるさいのである。ふふふ。

協力：オリンパス光学工業株式会社

⑨完成した合成画像

⑩ジェニーの絵が壁に描かれたビル

よいこのSX-WINDOW講座（第4回）

アイコンのドラッグとアイコン化

Nakamori Akira 中森 章

今回は，SX-WINDOWで特徴的なアイコンのドラッグのしくみについて考え，プログラムに組み込むまでを学びます。そしてそれらを理解したうえで，ウィンドウのアイコン化についても考えてみたいと思います。

SX-WINDOW上のアプリケーションを動かしていると，なかなか興味深い動作をするプログラムに出合うことがあります。たとえば，キャンバス.Xとかサウンド.Xといったプログラムは絵や音楽のデータファイルのアイコンをドラッグ（マウスの左ボタンを押しながら引きずること）してウィンドウに放り込むと，そのドラッグされたファイル名に応じて絵を表示したり音を鳴らしたりします。この動作を私たちのスケルトンプログラムに組み込んでやろうというのが今回のテーマです。さらに，1991年5月号のおまけディスク「黄金週間PRO-68K」以来Oh!X編集部が提唱している，ウィンドウのアイコン化についても対応してみましょう。

アイコンのドラッグのしくみ

アイコンのドラッグはSX-WINDOW上でのアプリケーションを扱ううえで特徴的な動作のひとつです。それは，“A:”とか“B:”とかいったドライブのウィンドウ内にあるファイルのアイコンをドラッグしてアプリケーションのウィンドウに放り込むと，それを入力ファイルと認識してアプリケーションの処理が始まるというものです（図1）。これは，SX-WINDOWでアプリケーションに対して入力ファイルを指定するための一般的なインタフェイスと考えられます。実際，ノート.X，サウンド.X，キャンバス.X，タイプ.X……と，このインタフェイスを採用しているアプリケーションは枚挙にいとまがありません。いうまでもなく，このインタフェイスの利点はファイル名をいちいちキーボードから入力しなくても，アプリケーションが必要とするファイル名をマウスで簡単に与えてやれることです。このアイコンのドラッグのしくみはどうなっているのでしょうか。これについて学んでいきましょう。

アイコンのドラッグを制御しているのはタスク管理を行うマネージャであるタスクマンです。タスクマンはアイコンのドラッグが始まると，移動中のマウスカーソルがある位置にあるウィンドウ（アプリケーション）に対して，「いまドラッグしているよ」とか「いまドラッグが終わったよ」というイベントを送りつけてきます。これを知っていればあとは簡単です。このタスクマンのイベントに対する処理をプログラムしてやれば，アイコンのドラッグ対応の出来上がりです。

具体的に説明しましょう。タスクマンのイベントは，

E_SYSTEM1

または，

E_SYSTEM2

という種類のイベントとしてアプリケーションに通知されてきます[1]。このときのイベントレコード（tseventという構造体）のwhat2フィールドに実際のタスクマンのイベントの種類が格納されています。これが，

DRAGNOW

であるときは，そのイベントが通知されたアプリケーションのウィンドウ上にドラッグ中のアイコンがあることを意味します。一方，タスクマンのイベントの種類が，

DRAGEND

であるときは，そのイベントが通知されたアプリケーションのウィンドウ上でアイコンのドラッグが終了した（マウスの左ボタンが離された）ことを意味します。したがって，これらのイベントが発生したときの処理を考えればよいのです。

1）タスクマンのイベントには「タスクの終了」，「全ウィンドウのクローズ」，「ウィンドウの選択」など現在30種類がある。アプリケーション側では少なくとも最初の3つをサポートしていないと，システムがまともに動作しない。

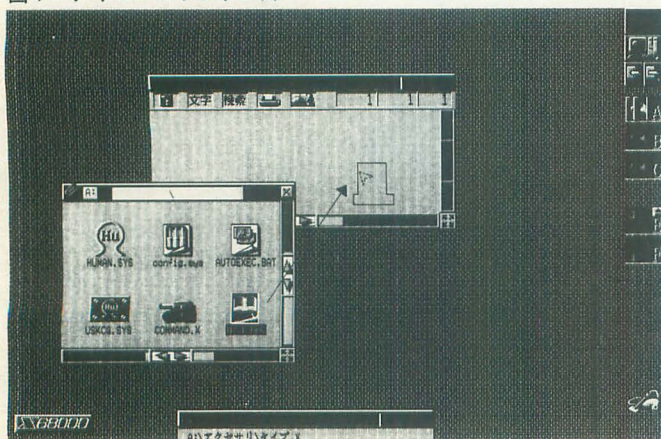
図1　アイコンのドラッグ例

アイコンのドラッグを通知されたら

それでは，アイコンのドラッグが発生したことを認識したとき，それらに対してどのような処理をしたらよいか，その例を説明します。

●DRAGNOW（ドラッグ中）に対する処理

このイベントに対する処理はアプリケーションによって大きく異なります。このイベントを受け取ったアプリケーションは，ウィンドウの表示を変更するなどの処理をして，ドラッグされているアイコンを受け入れる用意があることをアピール（？）しなければなりません。システムアイコンのクリーナ（掃除機）が，アイコンをドラッグしてきたときに吸い込み口を手前に向けるように形態を変えるのがその処理の例です。

ただし，多くのアプリケーションではこのイベントを無視することのほうが多いようです。したがって，スケルトンプログラムでも対処の必要はないのかもしれませんが，せっかくですから，ウィンドウの内枠を点滅させるくらいの処理をやらせてはどうでしょうか。ウィンドウへのポインタ

(WMOpenの戻り値）がwinPtrであるとき，

winPtr－＞wGraph.grRect

がウィンドウの描画可能なレクタングル（長方形の領域）を示しています。特殊な処理を行わない限りウィンドウの内枠のサイズはこのレクタングルと一致していますから，その枠を点滅させてやります。そのために便利な関数は，

GMInvertRect

です。この関数はレクタングルへのポインタと波線のパターンを引数とし，その波線のパターンで指定したレクタングルの外枠を描画します。実際の描画は波線パターンとのxor（排他的論理和）で行われるので，描画した枠をもとに戻すためには同じパターンを指定してもう一度GMInvertRectを呼び出すことが必要です。したがって，ウィンドウの内枠を点滅させるには，

GMInvertRect(&winPtr－＞wGraph.grRect, 0xf0f0)；
GMInvertRect(&winPtr－＞wGraph.grRect, 0xf0f0)；

などと，GMInvertRect関数を2個対にして呼び出せばよいのです。

●DRAGEND(ドラッグ終了)に対する処理

こちらのイベントへの対処は結構面倒です。アプリケーションに対してそのウィンドウ上でドラッグが終了したことが通知されてくるのですが，ファイル名などドラッグされたアイコンに関する情報は何もわかっていません。まず第1にそれらの情報を手に入れることが必要です。そのための関数が，

TSGetDrag

です。これはドラッグレコードと呼ばれるドラッグされたアイコンに関する情報へのポインタを得るための関数です。ドラッグレコードは図2のような構造をしています。このレコードにアイコンのファイル名などの情報が格納されています。

そして，ここで重要なのはセルリストです。セルリストに，具体的にドラッグされてきたアイコンに関する情報が入っているのです。セルリストの各セルの中のデータフィールドがその情報で，これはアイコン管理情報と呼ばれています[2]。アイコン管理レコードを得ることができれば，その次は，

TSISRecToStr

という関数を使ってファイル名を取り出すことができます。アイコン管理レコードにはファイルの属性とかアイコンのリソースIDなどの情報も含まれていますが，アプリケーションの立場からはファイル名以外は役に立ちそうにありませんね。データ型のキャストが複雑ですが，セルリストの先頭のセルに含まれているファイル名を取り出すための操作は，

```
セルへのポインタ＝＊(cell＊＊)
  (ドラッグレコードへのポインタ
  －＞cellHdl);
  TSISRecToStr(
  (icstate＊)
  (セルへのポインタ->data),
  ファイル名を格納する配列名)；
```

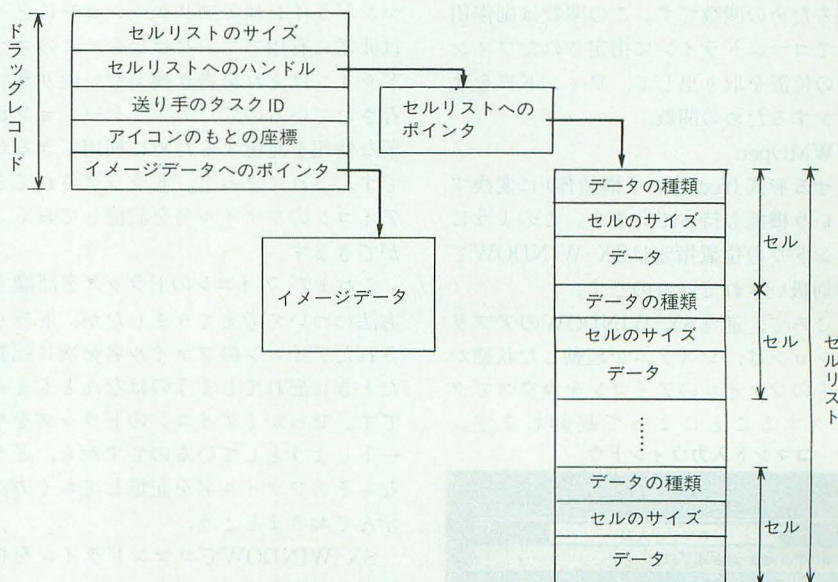
となります。

また，セルリストの2番目のセル(もしあれば）に含まれているファイル名は，セルの種類とサイズの情報に8バイトのデータが必要なことを考えると，

```
セルへのポインタ＝＊(cell＊＊)
  (ドラッグレコードへのポインタ
  －＞cellHdl);
セルへのポインタ＝(
  (char＊)セルへのポインタ
  ＋セルへのポインタ－＞length
  ＋8)；
TSISRecToStr(
  (icstate＊)(セルへのポインタ－＞
  data),ファイル名を格納する配列名)；
```

となるでしょう[3]。

セルへのポインタ－＞length ＋ 8の部分が先頭のセルのバイト数になるので，それをセルへのポインタに足し込んで次のセルの先頭を計算しているのです。

とにかく，ドラッグされたアイコンのファイル名がわかればしめたもの。あとはそのファイルをオープンして処理を開始するだけです。

ところで，ドラッグ中はアイコンの外形とファイル名を囲む点線がマウスで引きずられています。この点線はラバーバンドと呼ばれています。アイコンをドラッグするとき，このラバーバンドについて注意事項があります。つまり，ドラッグレコードを受け取ったあとは，

TSEndDrag

という関数によってラバーバンドを消してやる必要があります（このとき引数には0を指定します)。ドラッグをやめた位置でラバーバンドが消えるとその下にあるウィンドウにアイコンが吸い込まれていったように見えるので，その効果を生み出すためです。この関数を呼び出さない場合，あるいは引数に0以外を指定すると，アイコンがもとの位置まで戻るアニメーションが行われます。

---

2) セルレコードの種類としては，現在アイコン管理レコードと文字列がサポートされている。データの種類は4文字の文字列で表され，アイコン管理レコードは'FS'＋2バイトの文字，文字列は'STRN'という文字列である。ただし，ドラッグでアイコン管理レコード以外が渡されてくることがあるのかは不明（だれか教えて)。

3) セルへのポインタをchar＊にキャストしているのはバイト数を加算するため。そのまま加算したのではセルの大きさでスケーリングされた値（sxdef.hの定義によれば10倍の値）が加算されるため。本来ならセルへのポインタはchar＊よりもvoid＊にキャストすべき。XCのver.1でもコンパイルできるようにするためにchar＊にキャストしてある。

図2 ドラッグレコードの構造

ドラッグされたファイル名を記憶する

SX-WINDOWは再起動をしたとき(設定によっては)，前回の画面の状態(具体的には各ウィンドウの位置と大きさ)を覚えておいてそれを回復することができるようになっています。一見，当たり前のように思える機能ですが，これはSX-WINDOWのシステムとアプリケーションプログラムとの協力によって初めて実現されることなのです。すなわち，アプリケーションが起動するとき，もしそのアプリケーションが前回システムを終了したときに存在していたのなら，SX-WINDOWのシステムはそのアプリケーションのウィンドウが最後に存在していた位置の4角の座標（これで大きさもわかる）をコマンドラインに追加して渡してきます。このとき，アプリケーションはそのコマンドラインにウィンドウの位置指定があれば，優先してその位置にウィンドウをオープンする決まりになっているのです。システムは具体的には－wに続いて，

－w100,150,200,300

というような文字列をコマンドラインに追加してきます。この例では前回左上の座標(グローバル座標)が(100,150)，右下の座標が(200,300)である位置にウィンドウが存在していたことを意味しています。

ところで，

TSTakeParam

という関数をプログラムの中に見ることがありますが，これはコマンドラインを解析してコマンドラインにある引数のテーブルを作るための関数です。この関数は副作用としてコマンドラインに指定されたウィンドウの位置を取り出して，ウィンドウをオープンするための関数，

WMOpen

に渡せる形式(rectという構造体)に変換するという機能も持っています。このようにウィンドウの位置指定はSX-WINDOWでは特別扱いされているのです。

ところで，通常SX-WINDOWのアプリケーションは，システムが起動した状態から，そのファイルのアイコンをマウスでクリックすることによって起動します。

図3　コマンド入力ウィンドウ

Human68kのようにコマンドラインからコマンドを打ち込んでアプリケーションを起動するわけではないので，SX-WINDOWでコマンドラインといわれてもピンとこないかもしれませんね。実感を得るために，ここでちょっとした実験をしてみましょう。

アプリケーションはなんでもいいのですが，とりあえずSX-WINDOWのシステムディスクのアクセサリというディレクトリの下にあるタイプ.Xを使うことにします。SX-WINDOWのシステムが起動している状態で，キーボードのOPT.1キーを押しながらタイプ.Xのアイコンをマウスでダブルクリックします。すると，画面上に「コマンド入力」というタイトルのウィンドウが現れます。ウィンドウにはすでに，

A:1\1アクセサリ1\2タイプ

というクリックしたアイコンのファイル名が表示されていますね(図3)。このウィンドウには，文字を追加して入力できるようになっているので，

－w100,200,300,400

という文字をファイル名に続けて入力してリターンキーを押してみましょう。すると，グローバル座標で(100,200)と(300,400)である点を左上および右下の角とする位置にタイプ.Xのウィンドウがオープンします。「コマンド入力」ウィンドウこそ現れませんが，システムを再起動するときにはこれと同じようなことが内部的に行われているのです。これで，SX-WINDOWにもコマンドラインというものがあるということがなんとなくわかりますね。

さて，SX-WINDOWのこの内部的なコマンドラインはアプリケーションにとっては非常に有用です。なぜなら，このコマンドラインはそれを書き換えない限り常に保存されているので，アプリケーションが必要な情報を記憶するために利用できるからです。これによって，ドラッグされてきたアイコンのファイル名を記憶しておくことができます。

これまで，アイコンのドラッグを認識する方法について考えてきましたが，ドラッグされたアイコンのファイル名を次に起動したときに忘れてしまうのはなんともまぬけです。せっかくアイコンのドラッグをサポートしようとしているのですから，どうせならそのファイル名を記憶しておく方法も学んでおきましょう。

SX-WINDOWでコマンドラインを扱うために必要となるのは，

TSGetTdb

TSSetTdb

という2つの関数です。これは，本来はタスクの管理テーブル（taskという構造体で表される）を操作するための関数です。このタスク管理テーブルの中のcommandというフィールドが現在記憶されているコマンドラインなのです。これを書き換えてやるわけです[4]。ここで，TSGetTdbがタスク管理テーブルを取り出す関数，TSSetTdbがタスク管理テーブルに値を設定する関数です。

具体的にタスク管理テーブルにデータを設定するためには，次のような手順で行います。

```
task taskBuf; /＊これは宣言ね＊/
TSGetTdb(&taskBuf, －1)；
  /＊ taskBufにタスク管理テーブルをコピー＊/
taskBuf.command.Lstr
      ←コマンドラインの内容；
taskBuf.command.length
      ←コマンドラインの長さ；
TSSetTdb(&taskBuf, －1)；
  /＊ タスク管理テーブルに書き戻す＊/
```

ここでcommandというフィールドはLASCIIというデータ型なので長さ（バイト数）と内容（文字列）の両方を設定しています。実際にはcommand.Lstrへの代入は文字列の代入になりますからstrcpyなどの文字列操作関数を使用することになりますが，具体的にどうやるかはケースバイケースです。

ところで，(タスク管理テーブルの）コマンドラインの書き換えは，いつ行ってもいいというものではありません。ものごとにはタイミングというものがあります。SX-WINDOWでコマンドラインを変更するタイミングはタスクマンに指定してもらいます。すなわち，先のDRAGNOWとかDRAGENDと同様のタスクマンのイベントである，

SAVE

というイベントが発生したときにコマンドラインを書き換えることができます。

逆に，コマンドラインの参照はTSGetTdbによっていつでも行うことができます。しかし，一般にはアプリケーションが起動したときに，何よりもまずコマンドラインを参照して必要な情報を取り出してしまうことが多いようです。

4）コマンドラインではウィンドウの位置は考慮しなくてよい。前回のウィンドウの位置はアプリケーションが起動するときに自動的にコマンドラインに追加してくれる。

ウィンドウのアイコン化と復元

アイコンのドラッグに関する知識は，まあ，以上のようなものでしょうか。次はウィンドウのアイコン化について学ぶことにしましょう。ここでいうアイコン化とはウィンドウをタイトルバー（ドラッグリージョンともいう）だけの状態にしてしまうことです(図4)。X-Windowなどのウィンドウシステムに見られるような，ウィンドウが小さな四角形になってしまうようなものではありません。ウィンドウのアイコン化を，必要のないウィンドウをとりあえず目立たない形にして画面上に残しておくことと考えると，タイトルバーだけにするだけでも画面をすっきりとしたしたものにすることができます。

アイコン化の基本はウィンドウの縦方向の長さを単に0にしてやることです。このためには，

WMSize

という関数を使用します。この関数はウィンドウへのポインタ，ウィンドウのサイズを示す横方向と縦方向の大きさを組にしたpoint_t型のデータ，およびアップデートを行うか否かのフラグを引数とします。したがって，縦方向（y方向）の長さを0にした点のデータを引数として与えてやればよいのです。縦方向（x方向）は適当，というか，あらかじめ決めておいた大きさにすればよいでしょう。たとえば，

```
point_t pt;
```

というpoint_t型のデータがあるとすれば，

```
pt.p.x ← 横の大きさ
pt.p.y ← 0
```

という操作をした[5]あとに

```
WMSize(winPtr,pt,1)；
```

を実行してやるのです。これで，ウィンドウをタイトルバーだけにすることができます。

ウィンドウをアイコン化したら，次はもとに戻すことが必要です。この場合もWMSize関数を利用しますが，このためにはアイコン化する前のウィンドウの大きさを覚えておかなければなりません。先に述べたように，

```
winPtr->wGraph.grRect
```

がウィンドウの大きさを示していると思っていい（winPtrがウィンドウへのポインタとします）ので，アイコン化する前にこの大きさの情報を記憶しておけばよいでしょう。具体的にはwGraph.grRectというrect型の構造体のrightフィールドの値からleftフィールドの値を引いたものが横（x）方向の大きさ，bottomフィールドの値からtopフィールドの値を引いたものが縦（y）方向の大きさになります。このもとのウィンドウの大きさは，アイコンのドラッグのときのファイル名と同様に，適当な形式でコマンドラインに記憶しておきましょう。これにより，一度システムを終了し，そのあとにシステムを再起動してから，アイコン化されていたウィンドウ[6]を完全にもとに戻すことができるのです。

5）これは__POINT_Tを#defineで定義してある場合の話。そうでなければ，point_t型はlong型と同じになるので，

```
pt ←（横の大きさ << 16）+ 0
```

となる。このとき，上位16ビットが横，下位16ビットが縦の大きさである。

6）アイコン化されていたウィンドウでは起動時にシステムから通知されるウィンドウサイズの縦方向の大きさが0になる。逆に，これによりウィンドウが前回アイコン化されていたことを知ることができる。

ダブルクリックによるアイコンの復元

ウィンドウをアイコン化したりウィンドウに復元するためには，通常マウス右ボタンダウンイベントでのポップアップメニューでのアイテム選択で行います。このとき，アプリケーションでは，同一のアイテムを選択することによって，

アイコン　→ウィンドウ
ウィンドウ→アイコン

の変化を交互に切り替えることが多いようです。さて，欲をいえば，

ウィンドウ→アイコン

の変化（アイコン化）はポップアップメニューによってもよいのですが，

アイコン　→ウィンドウ

の変化はポップアップメニューではなく，簡潔にマウスのダブルクリックで行いたいものです。ということで，ついでにダブルクリックを認識する方法もここで説明しておきましょう。

ダブルクリックは次のようにして認識することができます。マウスの左ボタンが押されるたびにそのときの時間を記憶しておき，同時に前回マウスの左ボタンが押されたときの時間との差を計算します。この差が，SX-WINDOWのシステムが認識するダブルクリックの基準時間よりも小さければ，ダブルクリックが発生したと思うのです。

ダブルクリックの基準時間は，

EMDClickGet

という関数で参照することができます。種を明かせばこの程度の操作でダブルクリックを実現することができます。しかし，これをプログラムに取り入れるのは結構大変です。特に，インアクティブなアイコン化されているウィンドウをダブルクリックでウィンドウに戻すためには，1回目のクリックでウィンドウをセレクトし（アクティベートする），2回目のクリックでウィンドウに戻すという芸当も必要になります。図5にダブルクリックを認識するためのアルゴリズムの流れ図を示しておきます。あとで示す実際のプログラムを参考にしながら流れを追ってみてください。

新しいスケルトンプログラムに組み込む

それでは，これまで述べてきたアイコンのドラッグ，ウィンドウのアイコン化をスケルトン（骨格）プログラムに組み込みます。これをリスト1に示します。これは前々回の連載で示したスケルトンプログラムの拡張版です。ダイアログマンによる「……について」のダイアログを追加してあります。その他の変更・追加点は，

●procSYSTEM
DRAGNOW,DRAGEND,SAVEイベントに対する処理を追加

●progDRAG
DRAGNOW,DRAGENDの処理（新規）

●procMSLDOWN
ダブルクリックの認識でアイコン化されたウィンドウを復元する処理を追加

●procMSRDOWN
ポップアップメニューにウィンドウのアイコン化，復元を行うアイテムを追加

●toIcon
アイコン化の処理（新規）

●toWindow
アイコン化されたウィンドウの復元処理（新規）

●saveSize

図4　ウィンドウのアイコン化

a)アイコン化する前

b)アイコン化した後

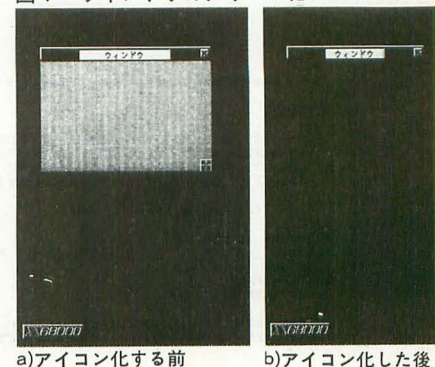
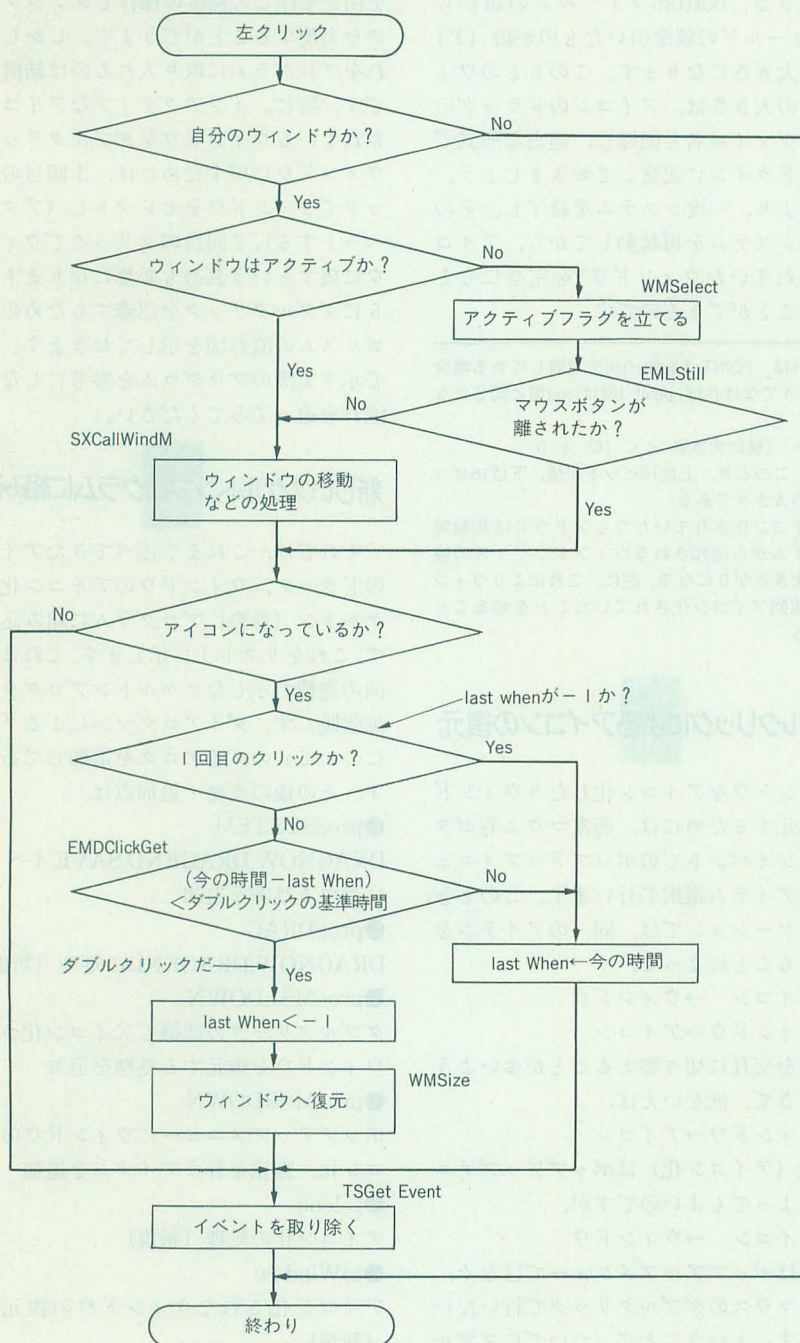



図5　ダブルクリックの認識

ドラッグされたアイコンのファイル名とアイコン化される前のウィンドウの大きさをコマンドラインに記録する（新規）

●recovSize

コマンドラインに記録されているファイル名とウィンドウの大きさを取り出す（新規）

といったところでしょうか。これまでの説明がわかっていればこれ以上の説明はいらないと思います。とにかくプログラムを打ち込んで動かしてみてください。なお，プログラムのコンパイルは，この連載の第1回，第2回目で紹介したバッチファイルを使用すると便利です。

おわりに

今回はスケルトンプログラムのバージョンアップを行ってみました。本当は，おまけディスクに付いていたSXEyesのように，初めて起動するウィンドウの大きさをマウスで指定できるようにしようと思って，そのプログラムも用意したのですが，紙面の都合で今回は掲載を見送らせてもらいました（リストの意味ありげな#undefがその名残）。

とにかく，これで一応まともなスケルトンプログラムができたので，次回からはこれを使ってSX-WINDOWのいろいろなマネージャを使う関数の動作を確かめていきたいと思っています。前にも少し話しましたが，この連載の目的はSX-WINDOWのドキュメントをやさしく解説するということです。やっと予告編が終わって本編が始まるというところでしょうか[7]。それでは，また。

7）かつて，予告編が死ぬほど長くて本編があっというまに終了した「聖闘士星矢」という作品もあったが……。

《参考文献》

1)吉沢正敏，SX-WINDOWプログラミング，ソフトバンク，1991年．

リスト

```
 1: /*
 2:
 3:      SX-WINDOWスケルトンプログラム Ver.2.0
 4:
 5: 【改版履歴】
 6:
 7:      May.29,1991 ダイアログマンを使用したダイアログを追加
 8:      May.29,1991 アイコン化をサポート（ダブルクリック対応）
 9:      May.31,1991 起動時のウィンドウサイズを可変に
10:      May.31,1991 ドラッグされたファイル名を記憶する
11:
12:           (C) 中森 章, May.31, 1991
13: */
14: #include <stdio.h>
15: #define __POINT_T   /* point_t 型を使う */
16: #include <sxlib.h>
17: #define FALSE    0
18: #define TRUE     ~FALSE
19:
20: #define EXT_GETWINSIZE  /* getWinSize() は外部で定義 */
21: #undef  EXT_GETWINSIZE  /* でも今回は内部で定義 */
22: #define ICON_WIDTH  210
23: /*
24:      ここでウィンドウに関する定数を設定
25: */
26: #define WDEFID       WI_STD
27: #define WINOPT       ( WC_GBOX | WC_GBOXON )
28: #define WINWIDTH     256
29: #define WINHIGHT     168
30: #define WINTITLE     "¥012ウィンドウ"
31: #define EVENTMASK    EM_EVERY
32:
33: #define MDEFID       1
34: #define MNENABLE     0xffffffff
35: #define MNITEMS      3
36: #define MNILIST      "¥0¥0¥021ダイアログを出す ¥0¥0¥013アイコン… ¥
0¥0¥1?"
37: #define MNTITLE      "¥014メニューだよ"
38: /*
39:      ここは定数から計算される定数
40: */
41: #define WINOPTL      ( WINOPT & 0xf )
```

```
 42: #define WINDEFID    ( WDEFID << 4 | WINOPTL )
 43:
 44: rect    getWinSize(int,int);
 45:
 46: window  *winPtr;
 47: rect    winSize;
 48: event   eventRec;
 49: int activeFlag;
 50:
 51: int     ctrlFlag;   /* コントロールがあるかないか */
 52: int menuFlag;   /* メニューがあるかないか */
 53: int iconFlag;   /* アイコンになっているかどうか */
 54: int lastWhen;   /* ダブルクリックの判定用 */
 55: point_t oldWinSize; /* アイコン時のウィンドウサイズ記憶用 */
 56: char fileName[90]="*";  /* ドラッグされたファイルネーム
 57:                 は1つのみ記憶しておく */
 58:
 59: menu    **menuHdl;
 60:
 61: menu theMenu = {
 62:     0,0,0,0,MNENABLE,0,{MNITEMS-1,MNILIST}
 63: };
 64:
 65: main()
 66: {
 67:     if( SX_init()==FALSE ) OpenError() ;
 68:     while( 1 ){
 69:         TSEventAvail(EVENTMASK,&eventRec);
 70:         switch( eventRec.eWhat ){
 71:         case E_IDLE:     procIDLE();      break;
 72:         case E_MSLDOWN:  procMSLDOWN();   break;
 73:         case E_MSLUP:    procMSLUP();     break;
 74:         case E_MSRDOWN:  procMSRDOWN();   break;
 75:         case E_MSRUP:    procMSRUP();     break;
 76:         case E_KEYDOWN:  procKEYDOWN();   break;
 77:         case E_KEYUP:    procKEYUP();     break;
 78:         case E_UPDATE:   procUPDATE();    break;
 79:         case E_ACTIVATE: procACTIVATE();break;
 80:         case E_SYSTEM1:  procSYSTEM();    break;
 81:         case E_SYSTEM2:  procSYSTEM();    break;
 82:         case E_USER1:    procUSER();      break;
 83:         case E_USER2:    procUSER();      break;
 84:         }
 85:     }
 86: }
 87:
 88: #ifndef EXT_GETWINSIZE
 89: rect
 90: getWinSize(xmin,ymin)
 91: int xmin,ymin;
 92: {
 93:     rect    r;
 94:     *(int *)&r.left = TSGetWindowPos();
 95:     r.right = r.left+xmin;
 96:     r.bottom= r.top +ymin;
 97:     return  r;
 98: }
 99: #endif
100:
101: SX_init()
102: {
103:     task    taskBuf;
104:     char    BUF[100];
105:
106:     TSGetTdb(&taskBuf, -1);
107:     if( (TSTakeParam(&taskBuf.command,&winSize,NULL,0,NULL
,NULL)&1)==0 ){
108:         iconFlag=FALSE;
109:         winSize=getWinSize(WINWIDTH,WINHIGHT);
110:         oldWinSize.p.x=winSize.right-winSize.left;
111:         oldWinSize.p.y=winSize.bottom-winSize.top;
112:     }
113:     else{
114:         recovSize();    /* ファイル名も同時に取り出す */
115:         if(winSize.top==winSize.bottom){
116:             iconFlag=TRUE;
117:         }
118:         else{
119:             iconFlag=FALSE;
120:             oldWinSize.p.x=winSize.right-winSize.left;
121:             oldWinSize.p.y=winSize.bottom-winSize.top;
122:         }
123:     }
124:     winPtr=WMOpen(NULL,&winSize,WINTITLE,TRUE,WINDEFID,(wi
ndow *)-1,TRUE,TSGetID());
125:     if( winPtr == NULL ) return( FALSE );
126:     winPtr->wOption = WINOPT;
127:     activeFlag=FALSE;
128:     lastWhen=-1;
129:     ctrlFlag  = CtrlPrepare();/* コントロールが不要なら ctrlFlag=
FALSE */
130:     menuFlag  = MenuPrepare();/* メニューが不要なら      menuFlag
=FALSE */
131:     drawGrowBox();
132:     return( TRUE );
133: }
134:
135: SX_term()
136: {
137:     if( ctrlFlag ) CtrlDispose();
138:     if( menuFlag ) MenuDispose();
139:     WMDispose( winPtr );
140:     exit();
141: }
142:
143: drawGrowBox()
144: {
145:     GMSetGraph( winPtr );
146:     WMDrawGBox( winPtr );
147: }
148:
149: CtrlPrepare()
150: {
151:     return( FALSE );
152: }
153:
154: CtrlDispose()
155: {
156:     return( FALSE );
157: }
158:
159: MenuPrepare()
160: {
161:     menuHdl=(menu**)MMChHdlNew( sizeof(theMenu) );
162:     if( menuHdl == NULL ) return ( FALSE );
163:     memcpy(*menuHdl,&theMenu,sizeof(theMenu));
164:     (*menuHdl)->mProc=RMRscGet(('M'<<24)|('D'<<16)|('E'<<8
)|'F',MDEFID);
165:     if( (int)((*menuHdl)->mProc)<=0 ){
166:         MMHdlDispose(menuHdl);
167:         return( FALSE );
168:     }
169: #if MDEFID==1
170:     (*menuHdl)->mHandle=MNTITLE;
171: #endif
172:     return( TRUE );
173: }
174:
175: MenuDispose()
176: {
177:     MMHdlDispose(menuHdl);
178:     return( TRUE );
179: }
180:
181: procIDLE()
182: {
183:     return( FALSE );
184: }
185:
186: procMSLDOWN()
187: {
188:     if( eventRec.eWhom != winPtr ) return( FALSE );
189:     if( activeFlag == FALSE ){
190:         WMSelect( winPtr );
191:         activeFlag = TRUE;
192:         if( EMLStill() == 0) goto checkDClick;
193:     }
194:     switch( SXCallWindM(winPtr,&eventRec) ){
195:     case W_INCLOSE:
196:         SX_term(); break;
197:     case W_INGROW:
198:     case W_INZMOUT:
199:     case W_INZMIN:
200:         GMClipRect(&winPtr->wGraph.grRect);
201:         break;
202:     }
203: checkDClick:
204:     if(iconFlag==TRUE){ /* アイコンになっている */
205:         if(lastWhen==(-1))
206:             lastWhen=eventRec.eWhen;
207:         else{
208:             if((eventRec.eWhen-lastWhen)<EMDClickGet()){
209:                 lastWhen=-1;
210:                 toWindow();
211:             }
212:             else
213:                 lastWhen=eventRec.eWhen;
214:         }
215:     }
216:     TSGetEvent(EVENTMASK,&eventRec);
217:     return( TRUE );
218: }
219:
220: toWindow()
221: {
222:     iconFlag=FALSE;
223:     WMSize(winPtr,oldWinSize,-1);
224: }
225:
226: toIcon()
227: {
228:     rect    r;
229:     point_t p;
230:
231:     iconFlag=TRUE;
232:     r=winPtr->wGraph.grRect;
233:     oldWinSize.p.x=r.right-r.left;
234:     oldWinSize.p.y=r.bottom-r.top;
235:     p.p.x=ICON_WIDTH;
236:     p.p.y=0;
237:     WMSize(winPtr,p,-1);
238: }
239:
240: procMSLUP()
241: {
242:     return( FALSE );
243: }
244:
245: procMSRDOWN()
246: {
247:     int item;
248:
249:     if( eventRec.eWhom != winPtr ) return( FALSE );
250:     GMSetGraph( winPtr );
251:     if( activeFlag == FALSE ){
252:         WMSelect( winPtr );
253:         activeFlag = TRUE;
254:         if( EMRStill() == 0){
255:             TSGetEvent(EVENTMASK,&eventRec);
256:             return( FALSE );
257:         }
258:     }
259:     item=MNSelect(menuHdl,eventRec.eWhere);
260:     TSGetEvent(EVENTMASK,&eventRec);
261:     switch(item){
262:     case 1:
263:         doDialog(); break;
264:     case 2:
265:         if(iconFlag==TRUE)
266:             toWindow();
267:         else
268:             toIcon();
269:         break;
270:     case 3:
271:         DMError(1,fileName);
272:         break;
273:     }
274:     return( TRUE );
275: }
276:
277: procMSRUP()
278: {
279:     return( FALSE );
280: }
281:
282: procKEYDOWN()
283: {
284:     return( FALSE );
```

```
285: }
286:
287: procKEYUP()
288: {
289:     return( FALSE );
290: }
291:
292: procUPDATE()
293: {
294:     if( eventRec.eWhom != winPtr ) return( FALSE );
295:     WMUpdate( winPtr );
296:     if( ctrlFlag ) CMDraw( winPtr );
297:     WMUpdtOver( winPtr );
298:     drawGrowBox();
299:     TSGetEvent(EVENTMASK,&eventRec);
300: }
301:
302: procACTIVATE()
303: {
304:     if( eventRec.eWhom == winPtr )    activeFlag = TRUE:
305:     else if( eventRec.eWhom != NULL ){
306:         if( activeFlag ) {
307:             activeFlag = FALSE;
308:             TSGetEvent(EVENTMASK,&eventRec);
309:         }
310:     }
311:     return( TRUE );
312: }
313:
314: procSYSTEM()
315: {
316:     switch( ((tsevent*)&eventRec)->what2 ){
317:     case CLOSEALL:
318:     case ENDTSK:
319:         SX_term(); break;
320:     case WINDOWSELECT:
321:         WMSelect( winPtr ); break;
322:     case DRAGNOW:
323:         procDRAG(DRAGNOW);
324:         break;
325:     case DRAGEND:
326:         procDRAG(DRAGEND);
327:         break;
328:     case SAVE:
329:         saveSize();
330:         break;
331:     }
332: }
333:
334: procUSER()
335: {
336:     return( FALSE );
337: }
338:
339: OpenError()
340: {
341:     DMError(0x101,"ウィンドウがオープンできません");
342:     SX_term();
343: }
344:
345: /************************************
346:   doDialog()  ダイアログを出す関数
347:  ************************************/
348: /*
349:     ここでダイアログウィンドウに関する定数を設定
350: */
351: #define DWINDEFID    (38<<4)
352: #define DWINTITLE    "¥020ダイアログだよん"
353: /*
354:      アイテムリスト
355: */
356: typedef struct dlgItem2 {
357:     long          dlgIHdl;
358:     rect          dlgIBounds;
359:     unsigned char   dlgIType;
360:     unsigned char   dlgISize;
361:     unsigned char   dlgIData[32];
362: } dlgItem2;
363:
364: struct {
365:     short     itemNo;
366:     dlgItem2 dItem1;
367:     dlgItem2 dItem2;
368:     dlgItem2 dItem3;
369:     dlgItem2 dItem4;
370: } dItemList ={
371:     4-1,
372:    {
373:     0,
374:     {256-8-42,128-8-18,256-8,128-8},
375:     DT_STDBTN,
376:     32,
377:     "¥007 O K "
378:    },
379:    {
380:     0,
381:     {4,4,252,16},
382:     DT_STCTXT+DT_DISABL,
383:     32,
384:     "¥001¥024このプログラムは…"
385:    },
386:    {
387:     0,
388:     {4,50,252,62},
389:     DT_STCTXT+DT_DISABL,
390:     32,
391:     "¥001¥036ウィンドウの骨格プログラムです"
392:    },
393:    {
394:     0,
395:     {240-96,80,240,92},
396:     DT_STCTXT+DT_DISABL,
397:     32,
398:     "¥377¥020中森 章 1991.5.31"
399:    }
400: };
401:
402: /*
403:       ダイアログを開く位置(中央よりも少し上にしてある)
404: */
405: rect dlBounds={ 384-128,256-64-20,384+128,256+64-20 };
406: /*
407:       フィルター関数
408: */
409: MyFilter(Dialog,ev)
410: dialog *Dialog;
411: event  *ev;
412: {
413:     point_t okbtn;
414:
415:     if( ev->eWhat == E_KEYDOWN ){
416:         if((short)(ev->eWhom)==13){
417:             okbtn.p.x=384+128-10;
418:             okbtn.p.y=256+64-20-10;
419:             ev->eWhere=okbtn;
420:             ev->eWhat =E_MSLDOWN;
421:         }
422:     }
423:     return 0;
424: }
425:
426: doDialog()
427: {
428:     dialog    *dialogPtr;
429:     dlgIList **dIHdl;
430:     int  ditem;
431:
432:     dIHdl=(dialog**)MMChHdlNew( sizeof(dItemList) );
433:     if( dIHdl == NULL ){
434:         DMError(0x101,"領域確保に失敗しました。");
435:         return ( FALSE );
436:     }
437:     memcpy(*dIHdl,&dItemList,sizeof(dItemList));
438:     dialogPtr=DMOpen(NULL,&dlBounds,DWINTITLE,TRUE,DWINDEF
ID,
439:             (window *)-1,TRUE,TSGetID(),dIHdl);
440:     if( dialogPtr == NULL ){
441:         MMHdlDispose(dIHdl);
442:         DMError(0x101,"ウインドウがオープンできません,");
443:         return( FALSE );
444:     }
445:     DMBeep(2);
446:     ditem=DMControl((void*)MyFilter );
447:     DMDispose(dialogPtr);
448:     MMHdlDispose(dIHdl);
449:
450: }
451:
452: /******************************************
453:   procDRAG() アイコンのドラッグを処理する
454:
455:       引数 DRAGNOW  ドラッグ中の処理
456:            DRAGEND  ドラッグ終了時の処理
457:  ******************************************/
458: procDRAG(type)
459: int type;
460: {
461:     drag    *dragPtr;
462:     cell    *cellPtr;
463:
464:     if( eventRec.eWhom != winPtr ) return( FALSE );
465:     switch(type){
466:     case DRAGNOW:    /* ウィンドウの外枠を点滅させるだけ */
467:         GMSetGraph(winPtr);
468:         GMInvertRect(&winPtr->wGraph.grRect,0xf0f0);
469:         GMInvertRect(&winPtr->wGraph.grRect,0x0f0f);
470:         GMInvertRect(&winPtr->wGraph.grRect,0xf0f0);
471:         GMInvertRect(&winPtr->wGraph.grRect,0x0f0f);
472:         break;
473:     case DRAGEND:    /* ドラッグレコードの先頭を取り込む */
474:         if(TSGetDrag(&dragPtr)<0){
475:             return( FALSE );
476:         }
477:         TSEndDrag(0);
478:         cellPtr=*(cell**)(dragPtr->cellHdl);
479:         if(((cellPtr->kind)&0xffff0000) == ('FS'<<16)){
480:             TSISRecToStr((icstate*)(cellPtr->data),fileNam
e);
481:         }
482:         break;
483:     }
484: }
485: /**********************************************************
486:   saveSize() コマンドラインにウィンドウのサイズと
487:              ドラッグされたファイル名(1つ)を書き込む
488:
489:   ウィンドウの位置はその後にシステムが書き込む
490:  **********************************************************/
491: saveSize()
492: {
493:     task    taskBuf;
494:     int len;
495:     int i;
496:     char    BUF[256];
497:
498:     sprintf(BUF,"-S%d,%d -F%s ",
499:         oldWinSize.p.x,oldWinSize.p.y,fileName);
500:     len=strlen(BUF);
501:     if(len>255) len=255;
502:     TSGetTdb(&taskBuf, -1);
503:     for(i=0;i<len;i++)
504:         taskBuf.command.Lstr[i]=BUF[i];
505:     taskBuf.command.length=len;
506:     TSSetTdb(&taskBuf, -1);
507: }
508: /*******************************************************
509:   recovSize() コマンドラインからウィンドウのサイズ
510:               とファイル名を取り出す
511:
512:   ウィンドウのサイズはアイコンになっているときに,
513:   元のウィンドウにに戻るとき参照する
514:  *******************************************************/
515: recovSize()
516: {
517:     task    taskBuf;
518:     int x,y;
519:     char    BUF[256];
520:
521:     TSGetTdb(&taskBuf, -1);
522:     sscanf(taskBuf.command.Lstr,"-S%d,%d -F%s %s",
523:         &x,&y,fileName,BUF);
524:     oldWinSize.p.x=x;
525:     oldWinSize.p.y=y;
526: }
```

吾輩はX68000である
［第4回］

魔法の函の正体は？

Izumi Daisuke　泉　大介

関数を忌み嫌う気持ちもわからぬではないが
理解してしまばこれほど便利なものはない
さあ，魔法の箱のからくりを解き明かそう

先月に引き続きＣ言語の概要をお届けする。先月，Ｃ言語はプログラムを制御するものだけを命令として残してほかはすべて関数にしてしまったと説明した。関数を境に数学が嫌いになった，という諸兄もいらっしゃるかと思うが，それは，

1）　教えるのが下手な教師にあたった
2）　さぼっていた

のいずれかの理由による。わかってしまえばこれほど簡単なものはない。そして，これほど便利なものもない。

不思議な函，関数

図１をご覧いただきたい。これは関数の概念を図示したものである。関数とは図にあるような魔法の箱で，原料を放り込むとバクッと加工してその結果を吐き出してくれる。何が出てくるかわからない魔法の箱というのもスリルがあっていいかもしれないが，それではちょっと実用にならない。そこでたとえば，ｘという原料を放り込めば，

$$x^2+3x+2$$

という結果が得られる関数，というように，その機能を明確にしておく必要がある。これならｘの代わりに２を放り込めば，

$$2^2+3\times2+2$$

という結果が得られるのだと類推できる。ついでに関数に名前もつけておけば，なお利用しやすい。

こうして出来上がったのが例の数学の教科書にある，

$$f(x)=x^2+3x+2$$

というあの表記なのである。＝の左側には関数の名前と放り込む原料が，＝の右側にはそのとき得られる結果が記されている。ｘの代わりに２を放り込めば，

$$f(2)=2^2+3\times2+2=12$$

となる。

原料はひとつとは限らない。２つの原料をもらってその平均を吐き出す関数ｇは，

$$g(x, y)=\frac{x+y}{2}$$

と書けるであろう。

●Ｃ言語と関数

Ｃ言語で使う関数も基本的には数学で使う関数とよく似ている。たとえばtoupperという関数は，原料として英字のASCIIコードを放り込めば，その英字の大文字のASCIIコードを吐き出す。放り込まれた英字が小文字でなかった場合には，放り込まれた英字のASCIIコードをそのまま吐き出す。Ｃ言語ではデバッガを使った場合と同じように，シングルクォート（'）で文字をくくるとASCIIコードに変換されるので，

```
toupper( 'a' );
```

とすれば，大文字のASCIIコード'A'が結果として吐き出されることになる。

原料を放り込めば結果を吐き出すこのような一般の関数を，Ｃ言語では少々拡張して「関数」と呼んでいる。原料を受け取らず結果だけを吐き出す関数もあれば，結果を吐き出すだけでなく加工している間に副作用を及ぼすものも存在するのだ。先月，数を画面に表示するのに使ったprintfも関数だが，この関数は原料を放り込むと「画面に文字を表示する」という副作用を及ぼす。結果として吐き出すのは表示した文字数である。そんなものを吐き出されても利用のしようがないわけで（もないが），副作用である「画面に文字を表示する」という機能が重視されるものの例といえよう。

Ｃ言語では自分で新たに

図１　関数の概念図

関数を作成することでプログラムを作っていく。たとえば画面に「こんにちは」と表示する関数を作って実行すれば，それは画面に「こんにちは」と表示するプログラムとなる。閏日を計算し，その答えを画面に表示する関数を作って実行すれば，閏日を表示するプログラムが出来上がる。すべてが万事この調子である。

●関数の作り方

ここで，関数の作り方について触れておこう。大雑把に見れば関数は，

```
吐き出す結果の型　関数名（　原料　）
原料の型宣言
{
    関数内で使う変数の宣言
    計算
    結果の吐き出し
}
```

という形をしている。ある整数の2乗を計算する関数を作るなら，

```
int sqr( x )
int x;
{
    int ans;
    ans= x*x;
    return ans;
}
```

となる。上との対比で眺めてみていただきたい。

セミコロン（;）のついている行とついていない行があるが，これは「文はセミコロンで終わる」というC言語のルールによっている。変数の宣言は，宣言文という文で行われる。したがって，

```
int x;
```

はセミコロンで終わっている。

```
int ans;
```

も同様である。次の，

```
ans= x*x;
```

というのは，式にセミコロンがつくことによって文となったものである。そして最後の，

```
return ans;
```

は結果を吐き出すための特別な文でリターン文と呼ばれる。これは命令のひとつで，

```
return 式;
```

という文法となる。ansと変数名を書いただけで式になるのだろうかと疑問を持たれるかもしれない。「変数×変数」が式なのは感覚的に納得しやすいが，変数名だけというのはどうも……，そう思われることだろう。しかし，C言語では変数名だけでも式となる。上の例では変数ansに答えを入れてからそれを返すという回りくどいことをしているが，returnの後ろに式がくることからもおわかりのように，実際には，

```
return x*x;
```

と単純に表記できる。もちろん，変数ansを宣言する必要もない。

複数の文を{}でくくったものは，複文と呼ばれる。複文は，文でありながらセミコロンで終わらない。したがって関数は，

```
吐き出す結果の型　関数名（　原料　）
原料の型宣言
複文
```

という形で宣言されるのだということもできる。最近では(ANSIの標準化によって)原料の型宣言を関数名に続くカッコ内で行うことができるようになっているので，

```
int sqr( int x )
{
return x*x;
}
```

とさらにコンパクトに表記することが可能だ。このとき原料の宣言には';'はつかない。

このようにして作成した関数は，

```
sqr( 2 ) + 3*2 + 2
```

のように式の中で使うことができる。式の計算には関数が吐き出した結果が使用されるので，これは，

$$2^2+3\times2+2$$

を計算するのと同じことである。関数fを

```
int f( int x )
{
    return sqr( x ) + 3*x + 2;
}
```

と宣言すれば，先に数学表記で示した関数f(x)のC言語版の出来上がりとなる。ちなみに関数に与える原料のことを「関数のパラメータ」という。また，関数が吐き出す結果のことを「関数の値」という。今後はこの呼び方でいくことにしよう。

最後に複文について補足しておこう。複文は複数の文を{}でくくったものだが，より正確には，

```
{
    複文の中で使う変数の宣言文
    宣言文以外の文
}
```

という構造をしている。複文の中で宣言された変数は，複文の中だけで有効となる。したがって，関数sqrの最初の例の中で宣言している変数ansは，この関数の中だけで有効な変数である。関数のパラメータはこの特殊な場合で，複文の外で宣言されてはいるがこちらもやはり続く複文の中だけで有効な変数として振舞う。したがって，上のように関数fと関数sqrで同じ名前のパラメータを指定しても，まったく影響されることはない。名前は同じでも，別の変数として扱われるのである。

●printf関数の使い方

画面に文字を表示する関数はいくつか用意されているが，最も簡単に便利に使えるのがprintf関数である。単純に文字列を表示するだけならば，

printf(文字列)；

のように書けばいい。

文字列型などという型はこれまでの説明になかったが，既出の型の簡単な応用で指定することができる。“A”などの文字にASCIIコードと呼ばれる数値が与えてあることはすでにお話しした。ASCIIコードは１バイトの数値，すなわち前回説明したchar型の数値である。文字列は文字をズラリと並べたものなので，これはそれぞれの文字のASCIIコードをズラリと並べたもので表現できることは容易に想像できよう。実際，文字列は図２のようにメモリに格納されている。これは先月閏日計算の結果をメモリに書き込んだ部分である。char型の数値が並ぶ文字列は，文字列の最後を示すのに０という数値を用いる。これによって文字列の最後はわかるので，文字列を指定するにはその先頭アドレスを示せばいい。char型のデータが入ったアドレス，すなわちcharへのポインタが文字列の正体である。

実際にプログラム中で文字列を指定する場合には，より簡単な方法が用意されている。文字列をダブルクォートで囲むのである。

"abcde"

と書いておけば，Ｃコンパイラは自動的にこれらの文字のASCIIコード(と最後の０)をメモリの適当な場所に格納し，その先頭アドレスを指定してくれる。

printf("Hello¥n");

のように，字面どおりの効果を内部で実現してくれるわけである。

上の例の最後についている「¥n」は改行を表す特別なマークで，エスケープシーケンスと呼ばれている。このほかによく利用されるものとして「¥t」が挙げられよう。こちらはタブを指定する。また，文字列中に‘ " ’を入れたい場合は，「¥"」と書けばいい。‘¥’はエスケープシーケンスを指定する特別な文字なので，‘¥’自身を表示するためには「¥¥」と２つ重ねて書く必要がある。

単純に文字列を表示する場合はこれでいいとして，printf関数が威力を発揮するのは，計算結果を文字列中に埋め込みたい場合である。この方法はちょっと変わっていて，まず，

“１年は○○日です”

という文字データを用意しておき，次に「○○」の位置に数値を埋め込んで文字データを完成するという方法をとるようになっている。

printf("１年は%d日です¥n", 365)；

なら，「%ｄ」の位置に365という整数が数字に直されて格納され，

１年は365日です

という文字列が画面に表示される。

printf("１年は%ｇ日です¥ｎ",
　　365＋(100－4＋1)／400.0)；

なら，%ｇの位置に365.2425という実数が数字に直されて格納され，

１年は365.2425日です

と表示される。「%～」というのは，そこにどんなデータを埋め込むかという指示である。埋め込むデータが整数ならば「%ｄ」を，実数ならば「%ｇ」を使用する。‘%’自身を表示するには，「%%」と２つ重ねる必要がある。ひとつだけでなく，複数のデータの埋め込みを指示することも可能である。もちろんこの場合には，埋め込むデータもその分だけ用意しなければならない。

●特別な関数main

前回，最も簡単なプログラムの形として，

```
void main( void )
{
    ……
}
```

を紹介したが，これはmainという名の特別な関数を定義しているところである。なにが特別なのかというと，プログラムの最初に実行される，という特徴を持っていることである。先ほど作成した関数fを使いたいなら，このmain関数から使えばいい。

```
void main( void )
{
  printf( "f (%d) = %dです¥n", 2 , f ( 2 ) )；
}
```

という調子である。コンパイルして実行すると，

ｆ(２)＝12です

と画面に表示される。関数が返す型のところにvoidと書いてあるが，これは「答えを返さない」という特別な型である。また，パラメータにもvoidと書いてあるが，これはパラメータがないという意味になる。答えを返さないため，先月紹介したmain関数にも，そして上のmain関数にもreturnはない。

●IOCSと文字列

吾輩のサービスルーチン集であるIOCSでも，ここに挙げたのと同様の方法で文字列を扱う。すなわち，文字を表すASCIIコード列＋０で文字列を表現するのである。図３-１をご覧いただきたい。これは文字列を表示するた

図２　メモリに収められた文字列

```
－ds　a 9900 a990f
000A9900　30　2E　32　34　32　35　00
          ↑　↑　↑　↑　↑　↑
          0　.　2　4　2　5
```

文字のASCIIコードを並べ，最後に０をつけたものが文字列として扱われる

めのサービス，No.21_Hを利用するプログラムである。文字列の先頭アドレスをA1にセットして利用する。ここでは$B0010_H$に文字列があるものとしてプログラムが作ってある。実際の文字列セットは，メモリにデータをセットするmesコマンドを使ってもいいのだが，ここではより簡単な方法を用いている。図3-2である。ここで用いている「dc」というのはMC68000の命令ではない。これは「メモリに～というデータをセットしなさい」という，マシン語への変換プログラムに対する指示である。

dc.b 10

なら10という1バイトのデータがセットされるし，

dc.l 10

なら10という1ロングワードのデータがセットされる。複数のデータをセットしたいなら，

dc.b 30, 28, 31, 30

などとカンマで区切って並べればいい。この調子でいくと文字列をセットするには，

dc. b 'A', 'B', 'C', 'D'

などとしなければならないところだが，さすがにこれは面倒なのでダブルクォートでくくって書く略記法が用意されている。図3-2はこれを使っている。文字列の最後を表す0は自動的には補われないので，自分で用意しなければならない。

これまで触れなかったが，自作したプログラムを眺めるには図4-1のようにすればいい。「l」コマンドに続けてプログラムの先頭アドレスを指定するのである。諸兄もプログラムの作り方にだいぶ慣れてきたと思うので，以後プログラムはこの形式で紹介することにしたい。これで，図3-1と比較するとずいぶんすっきりと，プログラムが見やすくなる。また図4-2は，さきほど「dc」でセットしたデータが本当にメモリに格納されているかどうかを確かめているところである。$B0010_H$以降にちゃんとセットされていることを確認されたい。

プログラムの実行はこれまでどおり，ブレイクポイントの設定，「g」コマンドの順に行う。

b0 b000e

g=b0000

で実行してみていただきたい。画面に文字列が表示されるはずである。

C言語の制御構造

関数の中の処理の流れを制御する命令を挙げておこう。関数の作り方，変数，そしてこの制御命令を覚えてしまえば，C言語の文法はほとんど卒業したと考えていい。それほどC言語の文法は簡素である。

図3 IOCSと文字列

```
1）プログラムの作成     -a b0000                              ←ここからプログラムを作る
                        000B0000  ori. b    #$00, D0
                                  movea. l  #$b0010, al       ←文字列の先頭アドレスをalにセット
                        000B0006  ori. b    #$00, D0
                                  move. l   #$21, d0          ←サービス番号をセット
                        000B000C  ori. b    #$00, D0
                                  trap      #15               ←サービスの実行
                        000B000E  ori. b    #$00, D0

2）文字列の設定         -a b0010
                        000B0010  ori. b    #$00, D0
                                  dc. b     "Hello, computer!", 0    B0010Hより文字列をセット
                        000B0022  ori. b    #$00, D0                 する。詳細は本文を参照
```

図4 セットしたプログラムを表示してみる

```
1）プログラムの表示     -l b0000 b000e                        ←いま作ったプログラムを表示
                        000B0000  movea. l #$000B0010, Al
                        000B0006  move. l  #$00000021,D0
                        000B000C  trap.    #$0F                ←ここまでがプログラム
                        000B000E  ori. b   #$65,D0

2）文字列の表示         -ds b0000 b0022

                        000B0000  22 7C 00 0B 00 10 20 3C 00 00 00 21 4E 4F 00 00   "|....  <...!NO..
                        000B0010  48 65 6C 6C 6F 2C 20 63 6F 6D 70 75 74 65 72 21   Hello, computer!
                        000B0020  00 00 00
                                  ↑下線部に文字列が入っている
```

●条件分岐

条件によって処理を分ける場合には，

```
if ( 条件1 )
  文1
else
  文2
```

という書式で指定する。これは，もし条件1が成立したら文1を実行しなさい。さもなければ（条件1が不成立なら）文2を実行しなさいという意味である。二者択一の場合に使用する。「さもなければ」が必要なければ，else以降は書かなくてよい。

三者，四者択一にしたいなら，「さもなくて，もし～なら」という方法を使う。これは，

```
else if ( 条件 )
  文
```

という表現を使う。これらの形式で書かれた条件分岐は，if文と呼ばれる。if文も文のひとつなので，複文の中で使うことができる。したがって，当然関数の中でも使用可能である。

条件は式で与える。もし式の値が0ならば条件は不成立である。0でなければ条件は成立する。したがって，

```
if ( 0 )
  printf( "条件1が成立¥n" );
else if ( 1 )
  printf( "条件2が成立¥n" );
else
  printf( "条件1・2ともに不成立¥n" );
```

というif文があれば，画面には「条件2が成立」と表示されることになる。

もちろん，足し算引き算などを駆使して0，1を作り出していたのでは表現できる条件は限られてしまうわけで，

1)　a > b　　a はbより大きい
2)　a < b　　a はbより小さい
3)　a >= b　a はb以上
4)　a <= b　a はb以下
5)　a == b　a はbと等しい
6)　a != b　　a はbと等しくない

などの演算が使える。これらの式の値は，関係が成立すれば1，不成立なら0である。したがって，

```
(5 < 3) * 10→0
(3 < 5) * 10→10
```

となる。

数学では，0以上3以下の数xを「0≦x≦3」と表記するが，C言語ではこのような表現は許されない。2つに分け，「0≦xかつx≦3」と表現しなければならないのである。このため，

7)　a && b　　aかつb
8)　a || b　　aまたはb

という演算も用意されている。

文は上のように式（ここでは関数呼び出しだけ）にセミコロンをつけたお馴染みの文のほかに，複文を使用することもできる。条件成立時の処理が複数ある場合には，複文にして指定する。これは以下の命令の説明でも同様である。

●閏年判定関数

練習を兼ねて，御仁がこれまでに作ったプログラムのなかから閏年かどうかを判定する関数をお届けしよう。関数を作る場合には，どのような関数にするかという仕様の決定が重要である。閏年の判定をするにしても，画面に「○○年は閏年です」と表示する関数もあれば，閏年なら1を，そうでなければ0を返す関数もある。御仁のプログラムは後者のプログラムである。これは御仁がC言語を使い始めた頃に作成したカレンダープログラム（御仁の得意技）から抜粋したためだ。この関数の判定結果によって，2月の日数を決定していたのである。

御仁の作成した関数では，西暦はパラメータとして受け取ることになっている。つまり，関数の書き出しは，

```
int check_uruu( int year )
```

である。あとは，前回の図の中で紹介したように，

1)　yearが4で割り切れれば閏年
2)　ただし，100で割り切れれば閏年ではない
3)　ただし，400で割り切れれば閏年

というルールで閏年の判定を行えばいい。

プログラミングを始めたばかりの人は，ここで途方にくれてしまうかもしれない。上の3つのルールをifで場合分けするのだということはわかっていても，実際どのように場合分けをするのかというところでつまずいてしまうのである。

まず第1の問題は，どうやって「割り切れる」かどうかを判定するかであろう。これは「割り切れる」とは「余りが0である」ことに気づくと，

　year % 4 == 0　（%は割り算の余りを求める）という式が使えることがわかる。

次に陥りがちな罠は，人間語の曖昧さにある。上の条件をそのまま，

```
int check_uruu( int year )
{
  if (year % 4 == 0)
    return 1;
  else if ( year % 100 == 0)
    return 0;
  else if ( year % 400 == 0)
    return 1;
  else
    return 0;
}
```

と，プログラムにするミスを犯してしまうのである。上

の条件は，なんとなくelse ifを使うとうまく表現できそうな気がするのだろう。しかし，気がするだけではプログラムは動かない。このif文を人間語に置き換えてみよう。

　　もしyearが4で割り切れるなら

　　　　閏年である

　　さもなくて，もしyearが100で割り切れるなら

　　　　…………

もう，おわかりだろう。4で割り切れなくて100で割り切れる年はないのである。

プログラムを作るためには，「ただし」などという曖昧な表現ではなく，「さもなくて，もし」を使って条件を立て直さなければならない。つまり，

1)　もしyearが400で割り切れるなら閏年
2)　さもなくて，もし100で割り切れるなら閏年でない
3)　さもなくて，もし4で割り切れるなら閏年
4)　さもなければ，閏年でない

この関数の制作は諸兄に任せるとしよう。

●複数のケースを扱う

else ifを使えば三者択一，四者択一を行うことができるが，多者択一を行うための命令として，

```
switch ( 式 ) {
  case 結果1 :
       文1
       ……
       break ;
  case 結果2 :
       文2
       ……
       break ;
  …………
  default :
       文n
       ……
       break ;
}
```

という形式も用意されている。これはswitch文と呼ばれる。まず式を計算し，その結果が結果1と等しければ文1……を，結果2と等しければ文2……を，……，いずれとも等しくなければ文n……を実行する。

caseの中では複数の文を書くことができるので，式の計算結果と一致したcaseの処理の最後を示すbreakを忘れてはいけない。これがなければ続けて次の文を実行してしまう。

if文のように自由に条件を記述はできないが，メニュー選択で押されたキーによって処理を分ける場合などにすっきりと表現できるのが，このswitch文である。

●繰り返し

同じ処理を延々と繰り返すのはコンピュータたる吾輩の得意技だが，このための制御命令も存在する。

```
while ( 条件 ) 文
```

は条件をチェックし，条件が成立していれば文を実行する。そして再び条件をチェック……と繰り返す。

```
repeat 文 while ( 条件 )
```

は，まず文を実行し，それから条件をチェックして成立していれば再び文の実行……と繰り返す。基本的にはC言語が用意している繰り返しはこの2つだけであり，それぞれwhile文，repeat文と呼ばれている。

たとえば1990～2000年の閏年を列挙するプログラムは，whileを使って次のように書くことができる。

```
void main( void )
{
   int   year;
   year = 1990;
   while ( year <= 2000 ) {
       if ( check_uruu( year ))
           printf( "%d年¥n", year );
       year = year + 1;
   }
}
```

このwhileを簡素に書く方法として以下の命令もある。

```
void main( void )
{
   int   year;
   for ( year=1990; year<=2000; year=year+1)
       if ( check_uruu( year ))
           printf( "%d年¥n", year );
}
```

対比して眺めていただきたい。どちらも動作は同じである。こちらはfor文と呼ばれている。

初心者の方は，

```
year = year + 1;
```

という表記に戸惑われるかもしれない。C言語では＝は「代入」の記号である。したがって上の式は，yearに1を加え，それを再びyearにセットするという意味になる。決してyearはyear+1と等しいという意味ではないので注意されたい（等しいは==）。

さらなる深淵へ

この2回の連載で，諸兄はとりあえずC言語のプログラムを作れるだけの知識を獲得した。

来月は，ポインタの復習を兼ねて，配列，構造体といったデータ型に触れ，C言語入門の終幕とする所存である。そして，当初の目的であったグラフィックの探求へと稿は進むのであった。

X68000用 ©SEGA
パワードリフトより **SIDE STREET** Shindoh Noriyuki 進藤 慶到

X68000用 ©日本ファルコム
ワンダラーズ・フロム・イースより **Be Careful!** Watanabe Kazuhiko 渡辺 一彦

X68000用 ©SEGA
TURBO OUTRUNより **Checker Flag** Nishimoto Hideki 西本 英樹

MIDI X68000用（要U20/U220 Musicdrv.x） ©SEGA
パワードリフトより **Artistic Traps** Kamomiya Atushi 鴨宮 淳

待ちに待ってた，やっときた。ゲームミュージックの大特集です。ストックからの選曲が多いのでちょっと古めですが，いつ聴いてもすごいものはスゴイのです。ゲームミュージックのA級作品，入力しないとソンするかもよ？

イキナリ進藤だぁ！

はっはっは。いわずと知れた進藤君です。7月号の149ページのハミダシは前フリだったんですね。ここまで読んで，もう入力を始めた人もいるかもしれません。このLIVE inでは有名ですが，彼のレベルは常人の域を超えているのです。知らなかった人は要チェック，夏休み明けの試験には必ず出ますからね（ウソ百分率・90%）。

おっと，曲の紹介がまだでしたね。SEGAの体感ゲーム・パワードリフトより，Aコースのテーマ，「SIDE STREET」です。いままでSEGAさんとはご無沙汰していた分，今月は一気に3曲となり，ちょっと見にはSEGA特集に見えるかもしれませんね。その中の第1弾は彼に飾ってもらいましょう。

相変わらず長い音色定義で，19音もあります。まあ，メタルホークは24音でしたけどね。リストも431行と，縮める努力がそこかしこに見受けられるのですが，なかなか短くはならないものですね。

オープニングから「おやっ？」と思わせるようなこのノリ。曲が始まるのに時間がかかるだけで進藤君を予感させますが，やっぱり進藤君だったのですね。この曲はメタルホークのように完全コンパチではありませんが，ポイントをきっちりと捕らえたアレンジっぽくなっています。アーケード版にもひけをとらないようなデキのよさは並大抵ではありません。原曲に必ずある独特の泥臭さがぬけて，スマートな曲として完成されていて，単に楽譜を打ち込むだけでは得られない独特の雰囲気を持っています。

FM音源だけでは中低音の厚みが弱いと感じられますので，イコライザーで500Hz前後を強めにしてみるともっとカッコよくなります。

エンドレスループではありませんので，最後まで楽しんでください。

パワードリフト

TURBO OUTRUN

ファルコムさん，いらっしゃい

一時期は毎月のように載っていたファルコムさんも本当にご無沙汰です。ワンダラーズ・フロム・イースより「Be Careful!」をお送りしましょう。これは最初の冒険であるティグレー採石場の曲ですね。あのエドガーさんが行方不明になっている場所です。

作者の渡辺君は，夢幻戦士ヴァリスIIの「Sacred Sacrifice」で一度お世話になっていますね。これはそのときに同封されていた曲なのですが，なかなかまとまったアレンジをしてあったのでよく覚えていました。

ヴァリスIIのときもそうだったのですが，この作品もヘヴィーメタルっぽいアレンジがされています。完全に趣味の世界といったところでしょうか。実はこの「〜っぽいアレンジ」といったところがポイントで，もともとメタルを意識して作曲されたと思われるワンダラーズ・フロム・イースの曲にマッチングがいいのです。

ところで実はこの作品，1年半も前に投稿されていたもので，当時はまだワンダラーズ・フロム・イースのX68000版は発売されていませんでした。この曲はX68000用のアレンジを予想して作っていただいたようですね。実際に発売されたワンダラーズ・フロム・イースではこんな感じになっていませんが，こちらもなかなかのデキ具合です。

個人的な意見としては，ワンダラーズ・フロム・イースのX68000版よりも迫力が2まわりも違うようなこちらのアレンジのほうが好きです。同じOPMを使いながら

も，ノリのよさや音の完成度は製品版をはるかに上回っています。パソコン用のゲームミュージックを投稿するのであれば，こういったアレンジを送っていただいたほうが掲載率があがるはずです。同じ音なら出ても当然なのですから。ちなみに，イースやイースIIの音にはこちらのほうが近いのではないでしょうか。

短めのリストですので，ワンダラーズ・フロム・イースを持っている人も持っていない人も，ぜひ入力してみましょう。

彼の作品のほとんどはボリュームが大きめになっています。ドラムに合わせたセッティングなのでしょうが，もう少しボリュームを絞ってもいいでしょう。

なぜかm_assignを2回もやっていたので，そこの1回分はカットさせていただきました。

タイトーの西本，あれ？

西本君といえばトップランディングのイメージが強かったですね。そこから私はタイトーの西本って思っていたのです。ところが前作はパワードリフトのEコース，「Artistic Traps」。そう，ちょうど今月号でもMIDI版が載っているあの曲だったのです。そして今回もまたまたSEGAのTURBO OUTRUNに挑戦してくれました。曲はエンディングテーマ「Checker Flag」です。これじゃあSEGAの西本君ですね。

やはり常連さんらしく，プログラムはシンプルでスマートな仕上がりになっています。とても好感が持てます。音色の解説などはありがたいものですよね。

原曲と聴き比べると，メロディシンセがきれいにまとまっていますね。いい換えれば，原曲のほうがシンセに力があるというか，ノイズが乗っているというか……。アタックをもう少し早めに設定してもいいかもしれませんね。それからベルが一定の音量で鳴っているように聞こえます。それでもOKなのですが，小さめのボリュームで鳴らしておいて，サビで少し前に出すといい効果が得られるのではないでしょうか。

この作品も約1年半も前に投稿していただいていたものです。西本君もさすがにボツだと思っていたでしょう。いい作品はストックとして大事にされている場合もあるんですよ。またいずれほかの作品でお世話になるハズですので，その際はよろしくお願いします（大胆な前フリ）。

これからもSEGAだけでなく，いろいろなメーカーにどんどん挑戦していってくださいね。　(S.K.)

パワーアップして再登場

さて，パワードリフトから「Artistic Traps」です。プログラムは，サン・ミュージカル・サービスのMusicdrv.x用で，楽器はローランドのU20/U220に対応しています。

作者の鴨宮さんはかなりのUの使い手らしく，音色の一部はなんとオリジナルです。それに伴って，楽器側のメモリを書き換えますので，それは困る！　という人はリストの頭の部分に書いてある注釈を参考に，Uの内部メモリをファイルに落としてください。必ずMIDIケーブルが，MIDIインタフェイス側のINと楽器側のOUTと接続されていることを確認してから行ってください。ちなみに，このように楽器のメモリ内容をファイルに落とすことは，バルクダンプ機能がある楽器ならばすべて可能です。今後MIDI対応プログラムが発表されていくに伴ってこういった手間は避けられないので，U以外のユーザーもこれを機に各自勉強しておきましょう。

ところで，この曲は以前にX68000の内蔵音源対応でこのLIVE inに載ったことがありました。本誌では基本的に同じ曲目は載せないらしいのですが，デキがよいものはそんな慣例にとらわれずバンバン載せていってくれるそうです（逆をいえば違う曲目を狙え！　ということでしょう）。

最後に注意！

MIDI対応のプログラムは一緒にそれを録音したテープがあると断然有利です。メモしておきましょうね。　(善)

LIVE質問箱

Q．6月号のナディアで，ボーカルの音色だけLFOのスピードが違うんですけど，なにか特別な理由はありますか。

編集部　(S.K.)

A．こちらの単なるミスです。できれば，すべての音色のスピードを208にしてください。　作者：加納&小原

Q．X1のMUSIC Basicが拡張できません。（中略）どうにかしてください。

塩瀬勇人（15）神奈川県

A．手紙だけでは状況がイマイチつかめなかったので，あなたの使っているシステム，MUSIC Basic，拡張のプログラム，演奏させたいプログラムなど，一式を編集部のLIVE担当者宛で送ってください。編集の都合上，2～3カ月以上かかるかもしれませんが，できるだけお答えしたいと思います。　(S.K.)

お知らせ

先日，あるプログラムコンテストに出品されたダライアスをアレンジしたプログラムが，タイトー社の許可を得ず，無断でパソケットで売られていたという事件がありました。このため，タイトー社では同社のゲームに関連したプログラムの雑誌掲載を許可できない状況となってしまいました。

本誌では，以前から各社のご好意によりミュージックプログラムその他を掲載させていただいていますが，この事態により，LIVE inでは今後タイトーのミュージックプログラムに関しては掲載できなくなりました。一部のユーザーの心ない行為でこのようなことになってしまったのは，非常に悲しいことです。また，今後こういった事件が何度も重なると，すべてのゲームに関するプログラムが扱えなくなる，という事態も起こりえます。そのような事態を招かないためにも，読者の皆さんもモラルと勇気を持って行動していただくようお願いします。（編集部）

リスト1　SIDE STREET

```
10 /*    save "SIDESTREET          .bas"
20 /*
30 /*      POWER  DRIFT
40 /*
50 /*     - SIDE STREET -
60 /*
70 /*        (C)SEGA
80 /*
90 /*  PROGRAMED BY ENG
100 /*
110 m_init()
120 key  3,"          @M
130 key  9,"m_stop()@M
140 key 10,"m_play()@M
150 /*
160 str p(33)[256]:char o(255),v(4,9),voi(4,10)
```

```
170 str b[256],c[256],d[256],e[256]
180 str hl[256],ml[256],ll[256],s[256]
190 /*
200 for i=1 to 8
210  m_alloc(i,7000)
220  m_assign(i,i)
230 next
240 /*
250 VD()
260 DD()
270 PD()
280 m_play()
290 end
300 /*
310 /*  SET MML TO TRACK
320 /*
330 func t(t)
340  r=0
350  while o(r)<>255
360   m_trk(t,p(o(r)))
370   r=r+1
380  endwhile
390 endfunc
400 /*
410 /*  VOICE SET
420 /*
430 func set(vn)
440  voi(0,0)=(v(4,1)*8)+v(4,0)
450  voi(0,1)=15
460  voi(0,9)=(vn=82)*2+(vn=77)+(vn=76)*2-(vn<>73)*2+1
470  for x=0 to 3
480   for y=0 to 9
490    voi(x+1,y)=v(x,y)
500   next
510  next
520 m_vset(vn,voi)
530 endfunc
540 /*
550 /*  ポルタメント
560 /*
570 func pol(pa,pb,pc,pd,pe)
580 str oto(11)[2]={"c","c+","d","d+","e","f","f+","g","g+","a
","a+","b"}
590   b="y"+str$(47+pd)+"," : c="" : x=pa+pb
600   for i=1 to pc
610     x=x-pb
620     if x<0   then { x=256+x
630       if pe=0  then pe=11:d=">" else pe=pe-1
640     }
650     if x>255 then { x=x-256
660       if pe=11 then pe=0 :d="<" else pe=pe+1
670     }
680     c=c+b+str$(x)+d+oto(pe)+"&":d=""
690   next
700   c=left$(c,len(c)-1)
710 endfunc
720 /*
730 /* DRUM DATA
740 /*
750 func DD()
760 hl="<{y55,64g&y55,192g-&y55,64g-&y55,192f&y55,64f}16>
770 ml="<{y55,64d&y55,192d-&y55,64d-&y55,192c&y55,64c}16>
780 ll= "{y55,64g&y55,192g-&y55,64g-&y55,192f&y55,64f}16
790 endfunc
800 /*
810 /*  VOICE DATA
820 /*
830 func VD()
840 /*
850 /*    AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  SYNTH 1
860 v={  31,  0,  2,  0,  0, 21,  0,  1,  0,  0,
870      31,  0,  0,  8,  0,  3,  0,  3,  0,  0,
880      31,  0,  0,  8,  0,  3,  0,  1,  0,  0, /* CON FBL
890      31,  0,  0,  8,  0,  3,  0,  1,  0,  0,          5,  7}
900 set(70)
910 /*
920 /*    AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  PIANO
930 v={  31,  0,  0,  0,  0, 32,  0,  8,  3,  0,
940      31, 15,  8,  7,  2,  0,  0,  8,  3,  0,
950      31,  0,  0,  0,  0, 25,  0,  4,  7,  0, /* CON FBL
960      31, 15,  8,  7,  2,  0,  0,  4,  7,  0,          4,  7}
970 set(71)
980 /*
990 /*    AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  TOM
1000 v={ 31,  0,  0,  0,  0,  0,  0, 15,  0,  0,
1010     31, 24, 14,  6,  5,  2,  0,  6,  0,  1,
1020     31, 17, 17,  8,  3,  0,  0,  1,  0,  0, /* CON FBL
1030     31, 17, 17,  8,  3,  1,  0,  2,  0,  0,          6,  7}
1040 set(72)
1050 /*
1060 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  C.HH
1070 v={ 31,  0,  0,  0,  0,  0,  0, 15,  0,  3,
1080     31,  0,  0,  0,  0,  0,  0, 15,  0,  3,
1090     31,  0,  0,  0,  0,  0,  0, 15,  0,  3, /* CON FBL
1100     31, 18, 13,  9,  7,  6,  0, 15,  0,  3,          0,  7}
1110 set(73)
1120 /*
1130 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  SYNTH 2
1140 v={ 31,  0,  0,  0,  0, 20,  0,  2,  3,  0,
1150     13,  0,  0,  7,  0,  7,  0,  2,  3,  0,
1160     31,  0,  0,  0,  0, 21,  0,  2,  7,  0, /* CON FBL
1170     13,  0,  0,  7,  0,  3,  0,  2,  7,  0,          4,  5}
1180 set(74)
1190 /*
1200 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  SYN BASS
1210 v={ 31, 11,  0,  3,  3, 29,  0,  1,  0,  0,
1220     21, 10, 10,  9,  2,  5,  0,  1,  0,  0,
1230     21, 10, 10,  9,  2,  5,  0,  1,  0,  0, /* CON FBL
1240     21, 10, 10,  9,  2,  1,  0,  2,  0,  0,          5,  7}
1250 set(75)
1260 /*
1270 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  RIDE 1
1280 v={ 31, 31,  0,  0,  1,  9,  0,  5,  0,  1,
1290     18, 31, 13,  5,  1,  1,  0, 11,  0,  2,
1300     31, 28,  5,  3,  3, 14,  0,  2,  0,  2, /* CON FBL
1310     21, 31, 13,  5,  4,  0,  0, 14,  0,  3,          4,  7}
1320 set(76)
1330 /*
1340 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  RIDE 2
1350 v={ 31, 31,  0,  0,  1,  9,  0,  5,  0,  1,
1360     31, 31, 14,  7,  1,  7,  0, 11,  0,  2,
1370     31, 28,  5,  3,  3, 14,  0,  2,  0,  2, /* CON FBL
1380     31, 31, 14,  7,  5,  0,  0, 14,  0,  3,          4,  7}
1390 set(77)
1400 /*
1410 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  D.GUIT 1
1420 v={ 31,  5,  0, 15,  3, 36,  0,  2,  7,  1,
1430     31, 12,  0, 15,  2, 30,  1, 11,  3,  0,
1440     31,  2,  0, 15,  3, 12,  0,  0,  7,  0, /* CON FBL
1450     31, 15, 10, 15,  2,  3,  1,  1,  3,  0,          3,  7}
1460 set(78)
1470 /*
1480 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  D.GUIT 2
1490 v={ 27, 15,  3,  1,  4, 28,  0,  2,  7,  1,
1500     26, 15,  7,  1,  5, 23,  1, 12,  3,  0,
1510     31,  2,  0,  1,  3, 11,  0,  0,  7,  0, /* CON FBL
1520     30, 14,  0,  7,  0,  4,  1,  1,  3,  0,          3,  7}
1530 set(79)
1540 /*
1550 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  SYNTH 3
1560 v={ 23,  3,  4,  3,  1, 30,  0,  8,  7,  0,
1570     15,  5,  0,  6,  1,  0,  0,  8,  7,  0,
1580     23,  3,  4,  3,  1, 25,  0,  8,  3,  0, /* CON FBL
1590     15,  5,  0,  6,  1,  0,  0,  8,  3,  0,          4,  7}
1600 set(80)
1610 /*
1620 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  SYNTH 4
1630 v={ 31, 22,  0,  0,  7,  8,  0,  5,  3,  0,
1640     31, 20,  0,  6,  6,  1,  1,  1,  3,  0,
1650     31, 22,  0,  0,  7,  6,  0,  5,  7,  0, /* CON FBL
1660     31, 20,  0,  6,  6,  1,  1,  1,  7,  0,          4,  4}
1670 set(81)
1680 /*
1690 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  O.HH 1
1700 v={ 31, 31,  0,  7,  1,  7,  0, 11,  0,  3,
1710     31, 28,  2,  7,  4, 16,  2, 12,  0,  3,
1720     31, 22,  0,  7,  4, 10,  1,  1,  0,  1, /* CON FBL
1730     31, 31,  8,  8,  0, 17,  2,  7,  0,  0,          1,  5}
1740 set(82)
1750 /*
1760 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  SAX
1770 v={ 21, 19,  1,  1,  1, 31,  0,  1,  0,  0,
1780     21, 12,  0,  1,  1, 27,  0,  1,  0,  0,
1790     21,  1,  0,  4,  1, 32,  0,  5,  0,  0, /* CON FBL
1800     24,  6,  4,  7,  2,  1,  0,  1,  0,  0,          0,  7}
1810 set(83)
1820 /*
1830 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  SOLO
1840 v={ 18,  0,  0,  5,  0, 33,  0,  4,  7,  0,
1850     16,  0,  0,  6,  0,  5,  2,  4,  3,  0,
1860     18,  0,  0,  5,  0, 43,  0,  4,  3,  0, /* CON FBL
1870     15,  0,  0,  6,  0,  2,  2,  4,  7,  0,          4,  7}
1880 set(84)
1890 /*
1900 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  O.HH 2
1910 v={ 31, 31,  0, 15,  1,  7,  0, 11,  0,  3,
1920     31, 28,  2, 15,  4, 16,  2, 12,  0,  3,
1930     31, 22,  0, 15,  4, 10,  1,  1,  0,  1, /* CON FBL
1940     31, 31,  8, 15,  0,  3,  2,  7,  0,  0,          1,  5}
1950 set(85)
1960 /*
1970 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  B.D
1980 v={ 10, 31,  0, 15, 15, 14,  0,  0,  4,  0,
1990     31, 31,  0, 15, 15,  6,  0, 15,  4,  0,
2000     31, 19,  0, 10, 15,  6,  0,  0,  4,  0, /* CON FBL
2010     31, 15, 11, 15, 15,  0,  1,  0,  4,  0,          1,  0}
2020 set(86)
2030 /*
2040 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  !!!
2050 v={  0,  0,  0, 15,  0,127,  0,  0,  0,  0,
2060      0,  0,  0, 15,  0,127,  0,  0,  0,  0,
2070      0,  0,  0, 15,  0,127,  0,  0,  0,  0, /* CON FBL
2080      0,  0,  0, 15,  0,127,  0,  0,  0,  0,          7,  7}
2090 set(87)
2100 /*
2110 /*   AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2  SYNTH 5
2120 v={ 31,  0,  0,  0,  0, 24,  0,  8,  3,  0,
2130     31,  0,  0,  9,  0,  0,  0,  8,  3,  0,
2140     31,  0,  0,  0,  0, 20,  0,  4,  7,  0, /* CON FBL
2150     31,  0,  0,  9,  0,  0,  0,  4,  7,  0,          4,  6}
2160 set(88)
2170 /*
2180 endfunc
2190 /*
2200 /*  PLAY DATA
2210 /*
2220 func PD()
2230 m_tempo(90)
2240 /*
2250 p(0)="@73o2q8@v127l16  y48,00 y3,3 y2,15rr
2260 p(1)="cy2,23cy2,15ccy2,23cy2,23cy2,15cc y2,23cy2,23cy2,15c
cy2,23cy2,23cy2,15cy2,23@82c@73
2270 p(2)="|:3"+p(1)+":|
2280 p(3)="cy2,23cy2,15ccy2,23cy2,23cy2,15cc y2,23@82cy2,23cy2,
15@73cy2,5ccy2,15cy2,15cy2,5c
2290 p(4)="cy2,23cy2,15ccy2,23cy2,23cy2,15cy3,2y2,12c y3,3y2,5@
82>ay2,5a@73<y2,15cy2,5ccy2,23c@72pl|:y2,15"+hl+":|
2300 p(5)="|:3y2,23@77<<c>>y2,23@73cy2,15@77<<c>>@73c:|@77<<c>>
y2,23@73cy2,15@77<<c>>@73c
2310 p(6)=p(5)+p(5)
2320 p(7)="|:3y2,23@77<<c>>y2,23@73cy2,15@77<<c>>@73c:|@77<<c>>
y2,23@73c@77<<c>>y2,15@73c
```

```
 2330 p(8)="y2,23@77<<c>>y2,15@72"+hl+"@77<<c>>@73cy2,15@72"+hl+
"@73c@77<<c>>y2,15@72"+hl
 2340 p(9)="@73cy2,23cy2,15@72"+hl+"ry2,15ry2,15ry2,15p2"+ml+"@7
3c
 2350 p(10)="cy2,23cy2,15ccy2,23cy2,23cy2,15cc y2,23cy2,23cy2,15
ccy2,23cy2,15cy2,15cy2,15c
 2360 p(11)="|:2y2,23@77<<c>>y2,23@73cy2,15@77<<c>>@73c:| y2,15c
y2,23cy2,15@82c@73cy2,15cy2,15cry2,23@82c@73
 2370 p(12)="@76y2,23<<dy2,23d>>@73cy2,15c32&y2,15c32y2,15cccc y
3,1y2,5ccy3,3y2,15cy3,2y2,5ccy3,3y2,23cy2,15cc
 2380 p(13)="@76y2,23<<dy2,23d>>@73cy2,15c32&y2,15c32y2,15@72p2"
+l1+"@73cc{y2,63c&y2,63c} y2,23@76<<d&|:3y2,23d&:|y8,0y2,23d&y2,
23dy2,23dy2,15@72>>"+ml
 2390 p(14)="@76y2,23<<dy2,23d>>@73cy2,15c32&y2,15c32y2,15cccc y
3,1y2,5ccy3,3y2,15cy3,2y2,5ccy3,1|:y2,11c:|y3,3y2,15@72"+ml
 2400 p(15)="@76y2,23<<dy2,23d>>@73cy2,15c32&y2,15c32
 2410 p(16)="|:y2,15@72p2"+l1+"@73c:||:5c:||:3y2,10c32&y2,10c32:
|
 2420 p(17)="y2,15@72p2"+l1+"y2,14@72p2"+l1+"@73cc@86o0q6l12y3,1
y2,5ay3,2|:y2,5a:|l16y2,5ary3,3y2,15ry2,23rq8o2@73
 2430 p(18)="|:7"+p(1)+":|@76<<@v114dy2,23@v118dy2,15@v121d@v123
dy2,23@v125dy2,23@v126dy2,15@v127dd y2,5cy2,5crry2,15>>>@72"+ml+
"<y2,15@85crr
 2440 o={ 0, 2, 3, 2, 4, 6, 6, 6, 7, 8, 9,
 2450      2, 1, 2,10, 6, 6, 6, 5,11,
 2460      2, 1, 2,10, 6, 6, 6, 5,11,
 2470     12,13,14,15,16,12,13,14,15,17,
 2480      2, 1, 2, 1, 2, 1, 2,10,
 2490      6, 6, 6, 7, 8, 9,
 2500      2, 1, 2,10,18,255}
 2510 t(1)
 2520 /*
 2530 /*
 2540 p(0)="@75o1q8 v12 l16  y49,40 ry2,5a&
 2550 p(1)="|:aa<ccddee>ggrr|1aara&:||2ffra&
 2560 p(2)="|:aa<ccddee>ggr|1raara&:||2a8.r":p(3)=p(2)+"r
 2570 p(2)=p(2)+"a&
 2580 p(4)="o1|:32f:||:32g:||:32f:||:16g:||:8e:|eedeea-ba&
 2590 p(5)="y8,1v13|:aarra4<d8.c4>raarra4g8.a4raarra4<d8.c4>raar
ra4|1<{ee-dd-cc-}2>:||2<{ee-d}4d-rr>v12a&
 2600 p(6)="|:aa<ccddee>ggrr|1aara&:||2aar
 2610 o={ 0, 1, 2, 1, 3, 4,
 2620      1, 2, 1, 3, 4,
 2630      1, 2, 1, 3, 4, 5,
 2640      1, 2, 1, 2, 1, 2, 1, 3,
 2650      4, 1, 2, 1, 2, 1, 1, 1, 6,255}
 2660 t(2)
 2670 /*
 2680 /*
 2690 p(0)="@79o3q8 v11 l16  y50,00 ra&|:
 2700 p(1)="aa@78aa@79a8@78aa@79gg@78gg@79aa@78a@79a&
 2710 p(2)="aa@78aa@79a8@78aa@79gg@78gg@79ff@78f@79a&
 2720 p(3)="aa@78aa@79a8@78aa@79gg@78g@79a8.|1@78a@79a&:|@72p2@v
127>|:"+l1+":|v13
 2730 p(4)="@79p1ff@78p1|:14f:|":p(4)="o2"+p(4)+p(4)
 2740 p(5)="@79p1gg@78p1|:14g:|@79p1gg@78p1gggggg@79p1gg@78p1ggg
@79p1g@78p1gg
 2750 p(6)="@79p1gg@78p1|:14g:|@79p1ee@78p1eeeeee@74o1v13e8deea-
b
 2760 p(7)="@79o3v12l16y50,16y2,5a&|:
 2770 p(8)="aa@78aa@79a8@78aa@79gg@78g@79a8.|1@78a@79a&:|rr<v14
 2780 p(9)="@79ff@78|:14f:|":p(9)="o2"+p(9)+p(9)
 2790 p(10)="@79gg@78|:14g:|@79gg@78gggggg@79gg@78ggg@79g@78gg
 2800 p(11)="@79gg@78|:14g:|@79ee@78eeeeee@74o1v13e8deea-b y50,0
0
 2810 p(12)="r@79o2v14p2
 2820 p(13)="|:aav9av8av14a4<d8.c4&v9c>v14aav9av8av14a4g8.a4&v9a
 v14aav9av8av14a4<d8.c4&v9c>v14aav9av8av14a4|1<{ee-dd-cc-}2>:||2
<{ee-d}4d-v9d-v8d-
 2830 p(14)="aa@78aa@79a8@78aa@79gg@78g@79a8.@78a@79a&:|
 2840 p(15)="|:
 2850 p(16)=p(8) : p(8)=p(8)+"y2,5
 2860 p(17)="v13"+p(4)+p(5)+p(4)+p(6)
 2870 p(18)="v12|:"+p(1)+p(2)+p(1)+"|1"+p(2)+":||2"+left$(p(1),l
en(p(1))-9)
 2880 o={ 0, 1, 2, 1, 3, 4, 5, 4, 6,
 2890      7, 1, 2, 1, 8, 9,10, 9,11,
 2900      7, 1, 2, 1, 8, 9,10, 9,11,12,13,
 2910      7, 1, 2, 1,14,15, 1, 2, 1,16,
 2920     17, 7, 1, 2, 1,14,18,255}
 2930 t(3)
 2940 /*
 2950 /*
 2960 p(0)="@79o2q8 v13 l16  y51,36 ra&|:
 2970 p(3)="aa@78aa@79a8@78aa@79gg@78g@79a8.|1@78a@79a&:|@72v12|
:"+l1+":|v13
 2980 p(4)="@79p2ff@78p2|:14f:|":p(4)="o2"+p(4)+p(4)
 2990 p(5)="@79p2gg@78p2|:14g:|@79p2gg@78p2gggggg@79p2gg@78p2ggg
@79p2g@78p2gg
 3000 p(6)="@79p2gg@78p2|:14g:|@79p2ee@78p2eeeeeer@74y51,44p2o1v
11e8deea-
 3010 p(7)="@79o2v14l16y51,36a&|:
 3020 p(8)="aa@78aa@79a8@78aa@79gg@78g@79a8.|1@78a@79a&:|rr
 3030 p(9)="@80@v112o1p3l16 |:c4.ara8.a8.ar:|g2g8.g8.grg4g4a8a8a
8a8
 3040 p(10)="|:c4.ara8.a8.ar:|f2b8.b8.brb4<b4@v113a8a8a8a
 3050 p(12)="r@79o3v14p1
 3060 p(16)=p(8)
 3070 p(17)="v13"+p(4)+p(5)+p(4)+p(6)
 3080 p(18)="v14|:"+p(1)+p(2)+p(1)+"|1"+p(2)+":||2"+left$(p(1),l
en(p(1))-9)
 3090 o={ 0, 1, 2, 1, 3, 4, 5, 4, 6,
 3100      7, 1, 2, 1, 8, 9,10,
 3110      7, 1, 2, 1, 8, 9,10,12,13,
 3120      7, 1, 2, 1,14,15, 1, 2, 1,16,
 3130     17, 7, 1, 2, 1,14,18,255}
 3140 t(4)
 3150 /*
 3160 /*
 3170 p(0)="@81o4q8 v12 l16  y52,12 p1 rr
 3180 p(1)="|:rcrrcrrr>bbrr<ccrr |1rcrrcrrr>bbrraarr<:||2rcrrcrr
r>bbr<crrrr
 3190 p(2)="@80@v110o1l16p2|:a4b4<c8.d8.e8>:|<d4c4>b8.<c8.>b8g2@
v111<c>ab<c@v112dcde
 3200 p(3)="|:f4g4a8.b8.<c8>@v113:|{b<c@v114ed}1f4e4@v115a-8ea-r
b8
 3210 pol( 0,-56,10,5,2):p(4)="y52,00@83o4v15@l2"+c+"&y52,00e46g
16b16
 3220 p(5)="g8&|:7y52,40g&y52,80g&y52,40g&y52,0g&y52,216g-&y52,0
g&:|y8,4
 3230 p(6)="e8d16
 3240 p(7)="e8&|:14y52,40e&y52,80e&y52,40e&y52,0e&y52,216e-&y52,
0e&:|y8,4
 3250 pol( 0,-50,12,5,0):p(8)="<"+c+"&y52,00d16d-32c32>
 3260 pol( 0,80,6,5,11):p(9)="b8&|:12y52,40b&y52,80b&y52,40b&y52
,0b&y52,216b-&y52,0b&:|"+c+"y52,00
 3270 pol( 0,20,12,5,7):e=c:p(10)="g8&"+e+"&y52,00f+16&
 3280 pol( 0,-48,6,5,6):p(11)=c+"&y52,00g8&@l1"+e+"y8,4@l2
 3290 p(12)="b8&|:5y52,40b&y52,80b&y52,40b&y52,0b&y52,216b-&y52,
0b&:|
 3300 pol( 0,68,6,5,11):e=c:pol( 0,-36,8,5,1):p(13)=e+"<"+c+"&y5
2,00@l20d@l2
 3310 p(14)="e8&|:3y52,40e&y52,80e&y52,40e&y52,0e&y52,216e-&y52,
0e&:|y8,4
 3320 pol( 0,-50,12,5,2):p(15)="o4@v123|:d4.c16&{c&c&y52,100c&y5
2,200cr}16"+c+"&y52, 0e16d8.|1>a8<:||2
 3330 p(16)="{c>brb}4g8&|:28y52,40g&y52,80g&y52,40g&y52,0g&y52,2
16g-&y52,0g&:|y8,4
 3340 pol( 0,-50,12,5,7):p(17)="g8"+c+"&y52,00a16
 3350 p(18)="g8&|:9y52,40g&y52,80g&y52,40g&y52,0g&y52,216g-&y52,
0g&:|y8,4g16&{g&g&y52,80g&y52,160g}16
 3360 p(19)=c+"&y52,00a8l8b4<c>b<cd16
 3370 pol( 0,-50,12,5,0):p(20)="r16@l2|:r2@88o2v10"+c+"&y52,00d1
6c4r16 r2>b8.<c4r16 r2
 3380 p(21)=c+"&y52,00d16c4r16 r2@71o2v13|1{ee-dd-cc-}2:||2{ee-d
d-rr}2
 3390 p(22)="r8   @84o2q8@v123l16 y52,12
 3400 p(23)="b1&b2&b&q7b4q8@v122{ag}abre2{dcde}8rdcr>b4..r{de}{f
gagab<cdcd-c-defga}2e8r4
 3410 p(24)="r{defg}8a32c8.&c32{ag} a4..{gd}{ree-d}8c8>{ageg}8a{
<c>a}
 3420 p(25)="l32<|:edc>a<:|cdedc>b<ce dcc+defgar.@v123egar.<de @
v124{e-dc>a}4<ee-dcc>a16<e-rre-rdc>arr8
 3430 p(26)="aa+b<c>bar4de-ab<crr8c>ag@v123agfdd+edc>@v122a8..e-
<e-r>@v121gagab<cd c>bab<dcdccdede@v122g@v123a4
 3440 p(27)="a-ge@v124a-ab<cde-ee-dcde-eg@v125a<@v126c8.>bg<c16>
bag<c>ba|:<c>bag:|bage<c>bage<c>ba@v125bgd@v124e-
 3450 p(28)="@v123edc>aga<c>a@v122gededcrr @v121rr>b16<ee-dcdega
@v122bagf @v124<ee-dcr>@v123bb-agrrrrr@v122ee-
 3460 p(29)="cc+dd+ee-dd->@v121b8.dg gagab<cdd+c>brb|:3<cdc>bab:
|<c>b <cded@v122e16g@v123a<@v124c>agg+aa+b<c@v125ee-dce-efg@v126
a4 @v0@87y52,12
 3470 p(30)=p(14)+"@v0@87o4v12l16y52,8@81p1
 3480 p(31)="|:3rcrrcrrr>bbrr<ccrr rcrrcrrr>bbrraarr<:||:rcrrcrr
r>bbrr<ccrr:|
 3490 o={ 0, 1, 1, 2, 3,
 3500      4, 5, 6, 7, 4, 5, 8, 9, 4, 5, 6, 7, 4,10,11, 8,12,13,1
4,
 3510     15,16,15,17,18,19,
 3520      4, 5, 6, 7, 4, 5, 8, 9, 4, 5, 6, 7, 4,10,11, 8,12,13,1
4,
 3530     15,16,15,17,18,19,
 3540     20,21,
 3550     22,23,24,25,26,27,28,29,
 3560      2, 3, 4, 5, 6, 7, 4, 5, 8, 9, 4, 5, 6, 7, 4,10,11, 8,1
2,13,
 3570     30,31,255}
 3580 t(5)
 3590 /*
 3600 /*
 3610 p(0)="@81o4q8 v12 @l1r y53,52 l16rr
 3620 p(2)="@80@v110o1l16p3|:a4b4<c8.d8.e8>:|<d4c4>b8.<c8.>b8g2@
v111<c>ab<c@v112dcde
 3630 p(3)=p(3)+"r12
 3640 pol(32,-56,10,6,2):p(4)="y53,32@83o4v11@l2"+c+"&y53,32e46g
16b16
 3650 p(5)="g8&|:7y53,72g&y53,112g&y53,72g&y53,32g&y53,248g-&y53
,32g&:|y8,5
 3660 p(7)="e8&|:14y53,72e&y53,112e&y53,72e&y53,32e&y53,248e-&y5
3,32e&:|y8,5
 3670 pol(32,-50,12,6,0):p(8)="<"+c+"&y53,32d16d-32c32>
 3680 pol(32,80,6,6,11):p(9)="b8&|:12y53,72b&y53,112b&y53,72b&y5
3,32b&y53,248b-&y53,32b&:|"+c+"y53,32
 3690 pol(32,20,12,6,7):e=c:p(10)="g8&"+e+"&y53,32f+16&
 3700 pol(32,-48,6,6,6):p(11)=c+"&y53,32g8&@l1"+e+"y8,5@l2
 3710 p(12)="b8&|:5y53,72b&y53,112b&y53,72b&y53,32b&y53,248b-&y5
3,32b&:|
 3720 pol(32,68,6,6,11):e=c:pol(32,-36,8,6,1):p(13)=e+"<"+c+"&y5
3,32@l20d@l2
 3730 p(14)="e8&|:3y53,72e&y53,112e&y53,72e&y53,32e&y53,248e-&y5
3,32e&:|y8,5
 3740 pol(32,-50,12,6,2):p(15)="o4      |:d4.c16&{c&c&y53,132c&y5
3,232cr}16"+c+"&y53,32e16d8.|1>a8<:||2
 3750 p(16)="{c>brb}4g8&|:28y53,72g&y53,112g&y53,72g&y53,32g&y53
,248g-&y53,32g&:|y8,5
 3760 pol(32,-50,12,6,7):p(17)="g8"+c+"&y53,32a16
 3770 p(18)="g8&|:9y53,72g&y53,112g&y53,72g&y53,32g&y53,248g-&y5
3,32g&:|y8,5g16&{g&g&y53,112g&y53,192g}16
 3780 p(19)=c+"&y53,32a8l8b4<c>b<cd16
 3790 p(20)="r16|:r6r4@88o1v10a8.a4r16 r2g8.a4r16 r2a8.a4r16 r2@
71o1v14|1{a-gg-fee-}2r12:||2{a-gg-frr}2
 3800 p(21)="
 3810 p(22)="r8r16@84o2q8@v117l16 y53,60 p2
 3820 p(23)="b1&b2&b&q7b4q8@v116{ag}abre2{dcde}8rdcr>b4..r{de}{f
gagab<cdcd-c-defga}2e8r4
 3830 p(25)="l32<|:edc>a<:|cdedc>b<ce dcc+defgar.@v117egar.<de @
v118{e-dc>a}4<ee-dcc>a16<e-rre-rdc>arr8
 3840 p(26)="aa+b<c>bar4de-ab<crr8c>ag@v117agfdd+edc>@v116a8..e-
<e-r>@v115gagab<cd c>bab<dcdccdede@v116g@v117a4
 3850 p(27)="a-ge@v118a-ab<cde-ee-dcde-eg@v119a<@v120c8.>bg<c16>
```

▶ 最近になってやっとメモリを増設した。これで2Mバイトだ。しかし，我がX68000は修理のため旅に出ていたのでした。早く戻ってきてくれ！　C compilerが泣いてるぞ。
鈴木　守人(17)愛知県

```
bag<c>ba|:<c>bag:|bage<c>bage<c>ba@v119bgd@v118e-
 3860 p(28)="@v117edc>aga<c>a@v116gededcrr @v115rr>b16<ee-dcdega
@v116bagf @v118<ee-dcr>@v117bb-agrrrrr@v116ee-
 3870 p(29)="cc+dd+ee-dd->@v115b8.dg gagab<cdd+c>brb|:3<cdc>bab:
|<c>b <cded@v116e16g@v117a<@v118c>agg+aa+b<c@v119ee-dce-efg@v120
a8.@v0@87y53,52
 3880 p(30)="e8&y53,72e&y53,112e&y53,72e&y53,32e&y53,248e-&y53,3
2er24 @v0@87o4v12l16y53,40@81
 3890 t(6)
 3900 /*
 3910 /*
 3920 p(0)="@81o3q8 v13 @l1r y54,28 l16p2 rr
 3930 p(1)="|:rarrarrrggrraarr |1rarrarrrggrrffrr:||2rarrarrrggr
arrrr
 3940 p(2)="@80@v108o1p3|:c4d4e8.f8.g8:|g4e4d8.d8.d8>b2@v109<ede
f@v110gfga
 3950 p(3)="|:a4b4<c8.d8.e8>@v111:|l4<de@v112gga-a-l16@v113a-8a-
a-ra-8.
 3960 p(4)="@v0@87o3v14@81p2
 3970 p(5)=p(1)
 3980 p(6)= "@80@v113o2p2l16 |:>a4.<erg8.f8.er:|d2e8.d8.erf4e4g8
f8e8f8
 3990 p(7)="|:>a4.<erg8.f8.er:|d2a8.g8.gra4a4@v114<e8e8c8>br
 4000 p(8)="@81o3v13aarr@70v12a4v11a8.a4r
 4010 p(9)="@81o3v14aarr@70v12<e4&v11d8.&c4r
 4020 p(10)="@81o3v14aarr@70v12<e4@81v14o3p2{a-gg-fee-}2
 4030 p(11)="@81o3v14aarr@70v12<e4@81v14o3p2{a-gg-frr}2
 4040 p(12)="v13o3|:3rarrarrrggrraarr rarrarrrggrrffrr:||:rarrar
rrggrraarr:|
 4050 o={ 0, 1, 1, 2, 3,
 4060       4, 5, 5, 6, 7,
 4070       4, 5, 5, 6, 7,
 4080       8, 9, 8,10, 8, 9, 8,11,
 4090       4, 5, 5, 5, 5,
 4100       2, 3, 4, 5, 5, 4,12,255}
 4110 t(7)
 4120 /*
 4130 /*
 4140 p(0)="@81o3q8 v13 l16  y55,00 y15,0 l16rr
 4150 p(2)="@80@v110o1p1|:f4g4g8.a8.b8:|b4a4g8.g8.g8d2@v111agga@
v112bab<c
 4160 p(3)="c4e4f8.g8.g8 @v113c4e4f8.g8.a8 l4gg@v114bbbbl16@v115
b8bbr<e8.
 4170 p(4)="@v0@87o4v14@81p1
 4180 p(5)="|:rcrrcrrr>bbrr<ccrr |1rcrrcrrr>bbrraarr<:||2rcrrcrr
r>bbr<crrrr
 4190 p(6)= "@80@v113o2p1l16 |:>f4.<crd8.c8.cr:|>b2b8.b8.brb4b4<
d8d8c8d8
 4200 p(7)="|:>f4.<crd8.c8.cr:|>b2<d8.d8.dre4e4@v114<e8e8c8>br
 4210 p(8)="@81o4v14eerr@70v12e4v11e8.e4r
 4220 p(9)="@81o4v14eerr@70v12a4&v11g8.&f4r
 4230 p(10)="@81o4v14eerr@70v12a4@81v14o3{ee-dd-cc-}2
 4240 p(11)="@81o4v14eerr@70v12a4@81v14o3{ee-dd-rr}2
 4250 p(12)="p2"+p(12)
 4260 t(8)
 4270 /*
 4280 /*    コンフィグレーションファイルは特に入力しなくても
 4290 /*   今までの物がそのまま使えます。
 4300 /*                                          - O.T.K. BAND -
 4310 endfunc
```

リスト2 Be Careful!

```
  10 /*
  20 /*        Wonders From Ys III
  30 /*
  40 /*
  50 /*              Be Careful!!
  60 /*
  70 /*
  80 /*
  90 /*                    For OPMA 90/01/15. Mon. Ippiko.W
 100 /*
 110 /*  CDの アレンジバージョンをもとに,かってにじぶんなりのアレンジをしちゃたのよよよーんんんんん
 120 /*                            (げんきょくと,たしょうちがいます)
 130 m_init()
 140 dim char v(4,10) : str m(99)[255],d(99)[99],sd1[10],sd2[10
],sd3[10],bd[10],ht[10],mt[10],lt[10]
 150   /*
 160   for i=1 to 8
 170       m_alloc(i,5000)
 180           m_assign(i,i)
 190       next  : m_alloc(9,32)
 200 /* Drums Set
 210 sd1="y2,14
 220 sd2="y2,17
 230 sd3="y2,21
 240  bd="y2,23
 250 /*
 260     ht="y2,24
 270     mt="y2,25
 280     lt="y2,27
 290 /* Elec. Tom
 300  d(0)=" {b&b-&a&a-&g&g-&f&e}8 "
 310  d(1)=" {b&b-&a&a-}16
 320  d(2)=" {b&b-}32
 330 /*
 340 d(10)=" {g&g-&f&e&e-&d&d-&c}8
 350 d(11)=" {g&g-&f&e}16
 360 d(12)=" {g&g-}32
 370 /*
 380 d(20)=" {e&e-&d&d-&c&>b&b-&a}8
 390 d(21)=" {e&e-&d&d-}16
 400 d(22)=" {e&e-}32
 410 /*
 420 d(30)=" {c&>b&b-&a&a-&g&g-&f}8
 430 d(31)=" {c&>b&b-&a}16
 440 d(32)=" {c&>b}32
 450 /* MML Data
 460 v={
 470 /*      A/F  OM  WF  SY  SP PMD AMD PMS AMS PAN nothing
 480          56, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
 490 /*       AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
 500          24, 16,  0,  3,  3, 26,  0,  3,  3,  0,  0,
 510          30,  0,  0,  3,  0, 27,  0,  1,  0,  0,  0,
 520          31,  0,  0,  5,  0, 28,  0,  1,  0,  0,  0,
 530          31,  0,  0,  8,  0,  0,  0,  1,  0,  0,  0}
 540   m_vset(70,v)               :/* Elec. Guiter
 550 /*
 560 v={
 570 /*      A/F  OM  WF  SY  SP PMD AMD PMS AMS PAN nothing
 580          57, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
 590 /*       AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
 600          31, 11,  0,  2,  2, 35,  2,  9,  0,  0,  0,
 610          31,  5,  0,  2,  1, 20,  1,  0,  0,  0,  0,
 620          31, 10,  5,  2,  1, 30,  1,  0,  0,  0,  0,
 630          31,  5,  2,  8,  1,  0,  1,  0,  0,  0,  0}
 640   m_vset(71,v)               :/* Bass
 650 /*
 660 v={
 670 /*      A/F  OM  WF  SY  SP PMD AMD PMS AMS PAN nothing
 680          59, 15,  2,  1,  0,  0,  0,  0,  0,  3,  0,
 690 /*       AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
 700          20,  0,  0,  2,  0, 16,  0,  7,  4,  0,  0,
 710          30, 12,  0,  5,  2, 20,  0,  5,  0,  0,  0,
 720          30, 12,  0,  5,  2, 25,  0,  4,  2,  0,  0,
 730          31,  0,  0,  9,  0,  0,  0,  3,  7,  0,  0}
 740   m_vset(72,v)               :/* Dist. Guiter
 750 /*
 760 v={
 770 /*      A/F  OM  WF  SY  SP PMD AMD PMS AMS PAN nothing
 780          62, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
 790 /*       AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
 800          31,  5,  5,  5,  2,  7,  0,  0,  1,  0,  0,
 810          31, 16, 11,  8, 13,  0,  0,  0,  0,  0,  0,
 820          31, 17, 10,  8, 12,  0,  0,  0,  0,  0,  0,
 830          31, 12, 10,  8, 12,  0,  0,  0,  0,  0,  0}
 840   m_vset(73,v)               :/* Tom Tom
 850 /*
 860 v={
 870 /*      A/F  OM  WF  SY  SP PMD AMD PMS AMS PAN nothing
 880          60, 15,  2,  0,  0,  0,  0,  0,  0,  3,  0,
 890 /*       AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
 900          31, 10,  8, 15, 15,  0,  0, 12,  3,  1,  0,
 910          31, 17, 10, 15, 15,  0,  0,  8,  3,  0,  0,
 920          31, 10,  8, 15, 15,  0,  0, 12,  7,  1,  0,
 930          31, 17, 10, 15, 15,  0,  0,  8,  3,  0,  0}
 940   m_vset(68,v)                :/* Hi-Hat close
 950 /*
 960 v={
 970 /*      A/F  OM  WF  SY  SP PMD AMD PMS AMS PAN nothing
 980          62, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
 990 /*       AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
1000          31, 10,  4,  4,  5, 40,  1,  8,  7,  0,  0,
1010          31, 12,  8,  6,  5,  0,  1,  1,  7,  0,  0,
1020          31, 10,  4,  6,  5,  0,  1,  1,  3,  0,  0,
1030          31, 10,  4,  6,  5,  0,  0,  1,  3,  0,  0}
1040   m_vset(74,v)                :/* Marimba (Release)
1050 /*
1060 v={
1070 /*      A/F  OM  WF  SY  SP PMD AMD PMS AMS PAN nothing
1080          60, 15,  2,  1,  0,  0,  0,  0,  0,  3,  0,
1090 /*       AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
1100          18,  7,  0,  4,  1, 30,  0,  0,  3,  0,  0,
1110          20,  0,  0,  7,  0,  0,  0,  2,  3,  0,  0,
1120          18,  7,  0,  4,  1, 15,  0,  0,  7,  0,  0,
1130          20,  0,  0,  7,  0,  0,  0,  2,  7,  0,  0}
1140   m_vset(74,v)               :/* voice
1150 /*
1160 v={
1170 /*      A/F  OM  WF  SY  SP PMD AMD PMS AMS PAN nothing
1180          61, 15,  2,  1,  0,  0,  0,  0,  0,  3,  0,
1190 /*       AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
1200          31,  0,  0,  4,  0, 24,  0,  2,  0,  0,  0,
1210          31,  5,  4,  7,  1,  0,  0,  1,  0,  0,  0,
1220          31,  5,  4,  7,  1,  0,  0,  1,  0,  0,  0,
1230          31,  5,  4,  7,  1,  0,  0,  1,  0,  0,  0}
1240   m_vset(75,v)               :/* PSG
1250 /*
1260 v={
1270 /*      A/F  OM  WF  SY  SP PMD AMD PMS AMS PAN nothing
1280          58, 15,  2,  1,  0,  0,  0,  0,  0,  3,  0,
1290 /*       AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
1300          25, 10,  0,  5,  1, 29,  1,  1,  1,  0,  0,
1310          25, 11,  0,  8,  2, 15,  1,  5,  1,  0,  0,
1320          28, 13,  0,  6,  5, 48,  1,  1,  0,  0,  0,
1330          14,  8,  0,  6,  0,  0,  1,  1,  1,  0,  0}
1340   m_vset(76,v)                 :/* Violin
1350 /*
1360 v={
1370 /*      A/F  OM  WF  SY  SP PMD AMD PMS AMS PAN nothing
1380          60, 15,  2,  1,  0,  0,  0,  0,  0,  3,  0,
1390 /*       AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
1400          28,  4,  4,  4,  2, 40,  0,  3,  3,  0,  0,
1410          22,  8,  7,  8,  0,  0,  0,  2,  3,  0,  0,
1420          28,  4,  4,  4,  2, 50,  0,  5,  7,  0,  0,
1430          22,  8,  7,  8,  0,  0,  0,  2,  7,  0,  0}
```

▶化学の実験中にクロロホルムが筆入れの中に……。1分後，中にあったものはすべてくっついてしまった。悲しい。
二宮　善弘(21)大阪府

```
1440   m_vset(54,v)               :/* Tube Bell 2
1450 /*
1460 v={
1470 /*      A/F  OM  WF  SY  SP PMD AMD PMS AMS PAN nothing
1480          59, 15,  2,  0,  0,  0,  0,  0,  0,  3,  0,
1490 /*       AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
1500          31,  0,  0,  3,  0, 40,  0,  0,  0,  0,  0,
1510          31,  0,  0,  3,  0, 20,  0,  8,  0,  0,  0,
1520          31,  0,  0,  3,  0,  0,  0,  0,  0,  0,  0,
1530          20,  0,  0,  8,  2,  0,  0,  2,  0,  0,  0}
1540   m_vset(90,v)              :/* Synth.
1550 /*
1560 m_tempo(112) :/* Track Data
1570 /*
1580 m(0)=" y48,32 @76 @v127 r8 l8 o3
1590 m(1)=" |:2 ra<e4d#4>b4 ra<e4d#2> ra<e4d#4>b4 ra<e4 |1 >a#2
 :| |2 <g4 @70 o4 v14 {g&g-&f&e&e-&d&d-&c&>b&b-&a}4
1600 m(2)=" v15 o2 l16 |:3 a>a<aa>a<aag#g>g<gg>g<ggg# :| a>a<aa
>a<aag#grrg&g8&{g&g-&f&e&e-&d&d-&c}8
1610 m(3)=" [do] |: ab<cdedc>ba>aa<a>aa<ag& gdgab<cd>bg>gg<g>gg
<gf& fgef& |1 f4abg#a&a8b<c defe&e>b<d>bg#8.<g#8.e8 :| |2 > fga8
<defe&e>b<dc > |:2 a>aa<a >aa<a>a< :|
1620 m(4)=" v12 @54 p1 o6 |:4 a<a>b<bc<c>>aa <a>ab<bc<c>>a<a> :
|
1630 m(5)=" v14 p3 o5 q6 f8.g8.e8f2 e8.f8.d#8e2 d4c4>b4a4 g#4.<
e8g#4 @70 o4 v15 o4 {g&g-&f&e&e-&d&d-&c&>b&a}4
1640 m(6)=" o2 |: >a<aa>a<aa>a<a <d>dd<e>ee<d>d< c>arar<cdc&c>a
rarara :| |:4 g>gg<g>gg<g>g< :| [loop]
1650 m_trk(1,m(0))
1660 m_trk(1,m(1))
1670 m_trk(1,m(2))
1680 m_trk(1,m(3))
1690 m_trk(1,m(4))
1700 m_trk(1,m(5))
1710 m_trk(1,m(6))
1720 /*
1730 m(0)=" y49,00 @76 v12 r4 o3 l8
1740 m(1)=" |:2 ra<e4d#4>b4 ra<e4d#2> ra<e4d#4>b4 ra<e4 |1 >a#2
 :| <g8 |2 @72 o4 v14 {g&g-&f&e&e-&d&d-&c&>b&b-&a}4
1750 m(4)=" v12 @54 p1 o5 |:4 a<a>b<bc<c>>aa <a>ab<bc<c>>a<a> :
|
1760 m(5)=" v14 p3 o5 q6 d8.e8.c8d2 c8.d8.>b8<c2 >b4a4f#4d#4 e4
.<c8e4 @72 o4 v15 q8 {g&g-&f&e&e-&d&d-&c&>b&a}4
1770 m_trk(2,m(0))
1780 m_trk(2,m(1))
1790 m_trk(2,m(2))
1800 m_trk(2,m(3))
1810 m_trk(2,m(4))
1820 m_trk(2,m(5))
1830 m_trk(2,m(6))
1840 /*
1850 m(0)=" y50,40 @75 o4 l4 r8 v10
1860 m(1)=" |: egf#d e2g2 egf#d e2 |1 a#2 :| |2 g2  |:3 a2 g2 :
| l16 a2grrg&g4
1870 m(2)=" [do] o5 |: e2cdefgfed& d2>bgb<cdegc& cd>b<c&cea8 |1
 fgef&fa<d8> b<c>ab&babge8.<e&e8g#8 :| |2 >fgag#&g#egb <c>baedc>
b<c>ab<ceab<c>a
1880 m(3)=" l8 > |: e.f16&f4e.d.c e4. |1 >a<c2 :| |2 ba2 > a2<f
.g.f a2c.d.e f2f#2g#2 l16 rebg<d>b<fe>
1890 m(4)=" |:16 a<aeg> :|
1900 m(5)=" g4b4<d4g4b4<d4g4b4 [loop]
1910 m_trk(3,m(0))
1920 m_trk(3,m(1))
1930 m_trk(3,m(2))
1940 m_trk(3,m(3))
1950 m_trk(3,m(4))
1960 m_trk(3,m(5))
1970 /*
1980 m(0)=" y51,00 @75 o4 l4 r8 v09
1990 m(1)=" |: ced#>b< c2d#2 ced#>b< c2d#2 :| |:3 c2d2 :| l16 c
2drrd&d4
2000 m(2)=" [do] o5 |: c2>ab<cdedc>b& b2gdgab<cd>a& abga&a<ce8
|1 decd&dfa8 g#afg&gege>b8.<b&b8<e8 :| |2 >defdd>b<eg edc>aedcdc
dea<cdec
2010 m(3)=" l8 |: c.c16&c4c.>b.a< c4. |1 >ea2< :| |2 ee2 >> r a
2<f.g.f a2c.d. v07 f2d#2 e2 l16 v09 rebg<d>b<fe>
2020 m(5)=" d4g4b4<d4g4b4<d4g4 [loop]
2030 m_trk(4,m(0))
2040 m_trk(4,m(1))
2050 m_trk(4,m(2))
2060 m_trk(4,m(3))
2070 m_trk(4,m(4))
2080 m_trk(4,m(5))
2090 /*
2100 m(0)=" y52,00 @75 p2 o4 l4 r8. v08
2110 m(1)=" |: egf#d e2g2 egf#d e2 |1 a#2 :| |2 g2  |:3 a2 g2 :
| l16 a2grrg&g4
2120 m(2)=" [do] o5 |: e2cdefgfed& d2>bgb<cdegc& cd>b<c&cea8 |1
 fgef&fa<d8> b<c>ab&babge8.<e&e8g#8 :| |2 >fgag#&g#egb <c>baedc>
b<c>ab<ceab<c>a
2130 m(3)=" > l8 |: e.f16&f4e.d.c e4. |1 >a<c2 :| |2 ba2 > a2<f
.g.f a2c.d.e f2f#2g#2 l16 rebg<d>b<fe>
2140 m(4)=" |:16 a<aeg> :|
2150 m(5)=" g4b4<d4g4b4<d4g4b4 [loop]
2160 m_trk(5,m(0))
2170 m_trk(5,m(1))
2180 m_trk(5,m(2))
2190 m_trk(5,m(3))
2200 m_trk(5,m(4))
2210 m_trk(5,m(5))
2220 /*
2230 m(0)=" y53,60 v13 @74 p1 o2 l8 r
2240 m(1)=" |:2 ra<e4d#4>b4 ra<e4d#2> ra<e4d#4>b4 ra<e4 |1 >a#2
 :| |2 <g2
2250 m(2)=" r1r1r1r1 [do] r1r1r1r1 r1r1r1r1
2260 m(3)=" @70 y53,32 o1 v15 l16 |: ff<f>f&ff<f>f&ff<f>fffgg a
a<a>a&aa<a>a&aa<a>a |1 aagg :| |2 aa<eg dddddddd<d>dd<d>d<d>d>
aaaaaaaa<a>aa<a>a<eg ddddd<d>ddd#d#d#d#d#<d#>d#d# eeeeeeee<e>e
e<e>ee<e> r1r1r1r1r1r1 [loop]
2270 m_trk(6,m(0))
2280 m_trk(6,m(1))
2290 m_trk(6,m(2))
2300 m_trk(6,m(3))
2310 /*
2320 m(0)=" y54,00 v13 @74 p2 o2 l8 r
2330 m(2)=" @71 o2 v15 l16 |:3 ar<a>ar<a>ag# gr<g>grgeg :| ar<a
>ar<a>ag#grrg&gceg
2340 m(3)=" [do] |: aa<a>aa<aeg>aa<a>a<a>aeg& gg<g>gg<gdf>gg<g>
g<g>gdf& ff<f>f&ff<fe |1 dd<d>d&d>ab<c eeeee<e>ee<e>ee<e>e>eg#b
:| |2 dd<d>deebg# aaaaaa<a>a<a>aa<a>aa<a>a
2350 m(4)=" |: ff<f>f&ff<f>f&ff<f>fffgg aa<a>a&aa<a>a&aa<a>a |1
 aagg :| |2 aa<eg dddddddd<d>dd<d>d<d>d> aaaaaaaaa<a>aa<a>a<eg
ddddd<d>ddd#d#d#d#<d#>d#d# eeeeeeeee<e>ee<e>ee<e>
2360 m(5)=" |:>a<aa>a<aa>a<a d<d>re<e>rd<d> c>aa<a>aa<a>a<c>a<a
>a<a>a<a>a :| |:4 <g>gg<g>gg<g>g :| [loop]
2370 m_trk(7,m(0))
2380 m_trk(7,m(1))
2390 m_trk(7,m(2))
2400 m_trk(7,m(3))
2410 m_trk(7,m(4))
2420 m_trk(7,m(5))
2430 /*
2440 m(0)=" @68 @v127 p2 y3,3 y2,8 r64 y2,8r16..l8
2450 m(1)=" y3,3 |:3"+bd+"r4 :|"+bd+"ry2,8r64y2,8r16..|:3"+bd+"
 r4:| y2,12r64y2,12r32. |:3 y2,12 r16 :|
2460 m(2)=" y2,11r64y2,62r16..y2,63r16y2,63r16 |:2 "+bd+"r4 :|"
+bd+"ry2,8r64y2,8r16..|:3"+bd+" r4:| y2,12r64y2,12r32. |:3 y2,12
 r16 :|
2470 m(3)=" y2,11r64y2,62r16..y2,63r16y2,63r16 |:2 "+bd+"r4 :|"
+bd+"ry2,8r64y2,8r16..|:3"+bd+" r4:| y2,11r64y2,11r32.y2,62r16y2
,63r16y2,64r16
2480 m(4)=" l16 |:3"+bd+"cc"+sd1+"c"+bd+"c c"+bd+"c"+sd1+"c"+bd
+"c :|
2490 m(5)=bd+"cc"+sd1+"c"+bd+"c c"+"y3,1 y2,62c32&y2,62c32 y3,3
 y2,63c32&y2,63c32 y3,2 y2,64c32&y2,64c32 y3,3
2500 m(6)=" y2,5 crr y2,5 c r y3,2 y2,62r y3,3 y2,63r y3,1 y2,6
4r y3,3
2510 m(7)=" y2,5 cc "+sd1+"cy2,99c"+bd+"c"+bd+"c"+sd1+"c"+bd+"c
 c"+bd+"c"+sd1+"cy2,99c"+bd+"c"+bd+"c"+sd1+"c"+bd+"c
2520 m(8)=" cy2,23c "+sd1+"cy2,99c"+bd+"c"+bd+"c"+sd1+"c"+bd+"c
 c"+bd+"c"+sd1+"cy2,99c"+bd+"c"+bd+"c"+sd1+"c"+bd+"c
2530 m(9)=" |:"+sd1+"c"+bd+"c"+bd+"c :|"+sd1+"c"+bd+"c"+"|:4"+s
d3+"r32 :| y3,1 |:4 y2,62 r32 :| y3,3 |:4 y2,63 r32 :| y3,2 |:4
y2,64 r32 :| y3,3
2540 m(10)="y2,5cc"+sd1+"cy2,99c"+bd+"c"+bd+"c"+sd1+"cy2,99c"+b
d+"cc"+sd1+"c"+bd+"c c"+bd+"c"+sd1+"cy2,99c
2550 m(11)="|:3"+bd+"cc"+sd1+"cy2,99c"+bd+"c"+bd+"c"+sd1+"cy2,9
9c"+bd+"cc"+sd1+"c"+bd+"c c"+bd+"c"+sd1+"cy2,99c :|
2560 m(12)="|:"+bd+"cc"+sd1+"cy2,99c"+bd+"c"+bd+"c"+sd1+"c"+bd+
"c "+bd+"c"+sd1+"c"+bd+"c"+bd+"c|1 y3,1 y2,62c32&y2,62c32 y2,62
c y3,3 y2,63 c y3,2 y2,64 c y3,3 :| |2 "+sd1+"c"+bd+"c"+sd1+"c"+
bd+"c
2570 m(13)="|:6"+bd+"c"+bd+"c"+sd1+"c"+bd+"c :|"+bd+"c"+sd1+"c"
+bd+"c"+bd+"c |:4"+sd3+"c32&"+sd3+"c32 :|
2580 m(14)="y2,5 cc"+sd1+"cy2,99c"+bd+"c"+bd+"c"+sd1+"c"+bd+"c
|:"+bd+"cc"+sd1+"cy2,99c"+bd+"c"+bd+"c"+sd1+"c"+bd+"c :|"+bd+"cc
"+sd1+"cy2,99c"+bd+"c"+bd+"c"+sd1+"c y2,8c64&y2,8c32.
2590 m(15)="|:3"+sd1+"r"+bd+"r"+bd+"r"+sd1+"r"+bd+"r"+bd+"r y2,
14 r y2,14 r :| |:16"+sd1+"r32 :|
2600 m_trk(8,m(0))
2610 m_trk(8,m(1))
2620 m_trk(8,m(2))
2630 m_trk(8,m(2))
2640 m_trk(8,m(3))
2650 m_trk(8,m(4))
2660 m_trk(8,m(5))
2670 m_trk(8,m(4))
2680 m_trk(8,m(6))
2690 m_trk(8,"[do]"+m(7))
2700 m_trk(8,m(8))
2710 m_trk(8,m(8))
2720 m_trk(8,m(8))
2730 m_trk(8,m(7))
2740 m_trk(8,m(8))
2750 m_trk(8,m(8))
2760 m_trk(8,m(9))
2770 m_trk(8,m(10))
2780 m_trk(8,m(11))
2790 m_trk(8,m(12))
2800 m_trk(8,m(13))
2810 m_trk(8,m(14))
2820 m_trk(8,m(14))
2830 m_trk(8,m(15)+"[loop]")
2840 print "Push Space bar to start!! " : while inkey$<>" "
2850     endwhile
2860         m_play(1,2,3,4,5,6,7,8)
2870     end
```

リスト3 Checker Flag

```
 10 /*
 20 /*         TURBO OUT RUN
 30 /*
 40 /*         ♪ Checker Flag ♪
 50 /*
 60 /*         PROGRAM ARRANGED by
 70 /*
 80 /*                 Hideki-Nishimoto.
 90 /*
100 m_init()
110 for I=1 to 8
120   m_assign(I,I)
```

▶ ドイツ語のやりすぎでBASICを「バーシック」と読んでしまう今日この頃。SAND BEIGEを「サンド・バイゲ」と読んでしまう方は重症です。気をつけましょう。

大塚　雅嗣(19)宮城県

```
130   m_alloc(I,1900)
140 next
150 /*
160 dim str X(32)[256]
170 str  S1[16], S2[16],  S[16]
180 str  B1[16], B2[16],  B[16]
190 str  C1[16], C2[16]
200 str  H1[16], M1[16], L1[16]
210 str   H[16],  M[16],  L[16]
220 str  R1[256],R2[256]
230 m_tempo(164)
240 /*
250 /*       Voice Data
260 /*
270 /*                  Bass
280 dim char V(4,10)={
290 /*  F/A  OM  WF SYC SPD PMD AMD PMS AMS PAN
300       58, 15,  2,  0,  0,  0,  0,  0,  0,  3,  0,
310 /*    AR D1R D2R  RR D1L  TL  KS MUL DT1 DT2 AMSE
320       31, 12,  4,  6,  6, 28,  2,  0,  3,  0,  0,
330       31, 14,  2, 14, 10, 48,  2,  4,  3,  0,  0,
340       31,  8,  3, 12,  5, 28,  2,  0,  3,  0,  0,
350       31,  8,  3, 12,  5,  0,  2,  0,  3,  0,  0}
360 m_vset(1,V)
370 /*                  Hihat(閉)
380 /*    オクターブ5のBで鳴らすととってもいいですよ。
390 /*   8分音符以下でOPEN,Q1にするとCLOSEにもなるお得品。
400 V={
410 /*  F/A  OM  WF SYC SPD PMD AMD PMS AMS PAN
420       59, 15,  2,  0,  0,  0,  0,  0,  0,  3,  0,
430 /*    AR D1R D2R  RR D1L  TL  KS MUL DT1 DT2 AMSE
440       31,  5,  4,  2,  2, 21,  0,  3,  3,  3,  0,
450       31,  3,  2,  1,  2, 22,  0,  1,  0,  2,  0,
460       31, 13, 11,  2,  2, 28,  0, 14,  0,  3,  0,
470       31, 19,  8,  6,  3,  0,  0, 14,  7,  3,  0}
480 m_vset(2,V)
490 /*                  Bell(PANに注意!)
500 V={
510 /*  F/A  OM  WF SYC SPD PMD AMD PMS AMS PAN
520       59, 15,  2,  0,  0,  0,  0,  0,  0,  2,  0,
530 /*    AR D1R D2R  RR D1L  TL  KS MUL DT1 DT2 AMSE
540       31,  7,  2, 11,  2, 67,  0,  2,  0,  0,  0,
550       31, 12,  2, 11,  7, 32,  0,  8,  7,  0,  0,
560       31, 10,  2, 11, 10, 54,  0,  1,  3,  0,  0,
570       31,  7,  8, 11, 10,  0,  1,  1,  3,  0,  0}
580 m_vset(3,V)
590 /*                  Bell(上と2箇所だけ違います)
600 V={/*                                          ↓ココダョ
610 /*  F/A  OM  WF  SY SPD PMD AMD PMS AMS PAN
620       59, 15,  2,  0,  0,  0,  0,  0,  0,  1,  0,
630 /*    AR D1R D2R  RR D1L  TL  KS MUL DT1 DT2 AMSE
640       31,  7,  2, 11,  2, 68,  0,  2,  0,  0,  0,
650       31, 12,  2, 11,  7, 32,  0,  8,  7,  0,  0,
660       31, 10,  2, 11, 10, 54,  0,  1,  3,  0,  0,
670       31,  7,  8, 11, 10,  8,  1,  1,  3,  0,  0}
680 m_vset(4,V)/*                 ↑ココモダイ
690 /*                  Synth 1(メロディ)
700 V={
710 /*  F/A  OM  WF  SY SPD PMD AMD PMS AMS PAN
720       60, 15,  2,  0,  0,  0,  0,  0,  0,  3,  0,
730 /*    AR D1R D2R  RR D1L  TL  KS MUL DT1 DT2 AMSE
740       12, 12,  2,  8,  1, 21,  1,  2,  3,  0,  0,
750       13, 10,  2,  8,  2,  2,  1,  2,  3,  0,  0,
760       12, 13,  2,  8,  1, 21,  1,  1,  7,  0,  0,
770       13, 10,  2,  8,  2,  0,  1,  1,  7,  0,  0}
780 m_vset(5,V)
790 /*                  Synth 2(ブラスI)
800 V={
810 /*  F/A  OM  WF  SY SPD PMD AMD PMS AMS PAN
820       58, 15,  2,  0,  0,  0,  0,  0,  0,  3,  0,
830 /*    AR D1R D2R  RR D1L  TL  KS MUL DT1 DT2 AMSE
840       31,  7,  6,  4,  5, 24,  1,  1,  3,  0,  0,
850       31,  7,  6,  4,  5, 40,  1,  1,  3,  0,  0,
860       31,  7,  6,  4,  5, 24,  1,  1,  3,  0,  0,
870       18,  6,  6,  2,  4,  0,  2,  1,  7,  0,  0}
880 m_vset(6,V)
890 /*                  Synth 3(ブラスII)
900 V={
910 /*  F/A  OM  WF  SY SPD PMD AMD PMS AMS PAN
920       58, 15,  2,  0,  0,  0,  0,  0,  0,  3,  0,
930 /*    AR D1R D2R  RR D1L  TL  KS MUL DT1 DT2 AMSE
940       14, 13,  0,  2,  1, 26,  3,  1,  3,  0,  0,
950       14, 13,  0,  1,  7, 39,  2,  1,  3,  0,  0,
960       14, 13,  0,  3,  2, 24,  2,  1,  3,  0,  0,
970       20,  3,  3, 10, 10,  0,  1,  1,  7,  0,  0}
980 m_vset(7,V)
990 /*
1000 /*         Set Drums
1010 /*
1020 /*[ SNARE DRUM ]
1030 S1="Y2,15B"
1040 S2="Y2,15BB"
1050 S="Y2,15"
1060 /*[ BASS DRUM ]
1070 B1="Y2,23B"
1080 B2="Y2,23BB"
1090 B="Y2,23"
1100 /*[ TOM TOM ]
1110 H1="Y3,1 Y2,59R Y3,3"
1120 M1="Y2,60R"
1130 L1="Y3,2 Y2,61R Y3,3"
1140  H="Y3,1 Y2,59 Y3,3"
1150  M="Y2,60"
1160  L="Y3,2 Y2,61 Y3,3"
1170 /*[ CYMBAL ]
1180 C1="Y2,3"
1190 C2="Y2,5"
1200 /*[ RYTHM ]
1210 R1=B2+S2+B1+B1+S2
1220 R2=B1+"R"+S1+"R"+B1+B+"R"+S1+"R"
1230 /*
1240 /*         Voice cheneg
1250 /*
1260 dim char V1(4,10)
1270 m_vget(4,V1)
1280  V1(4,5)=0
1290 m_vset(8,V1)
1300 /*
1310 /*         Music Data
1320 /*
1330 /*
1340 /*         < Hihat & PCM Drums >
1350 /*
1360 X(0)="Y3,3Y48,20 V15  L8 Q1 @2 O5"+C1+"R4"+C2+"R4"
1370 X(1)=C1+" BB "+S2+B1+B1+S2+"|:6 "+R1+" :|"
1380 X(2)=S1+S1+S1+S1+M1+M1+S1+L1
1390 X(3)=C1+" BB "+S2+B1+B1+S2+"|:5 "+R1+" :|"
1400 X(4)=B1+S+" R "+S+" BR"+S1+S+" RB "+S+"R"
1410 X(5)="|:2 "+S+" B "+S+"R :| L16"
1420 X(6)=H+" B "+H1+H1+H1+M+" B "+M1+L1+L1+"L8"
1430 X(7)=C1+" BR "+S+" BR "+B1+B+" R "+S1+" R|:6 "+R2+" :|"
1440 X(8)=B1+"L16"+H+"R4"+M1+M1+C1+"R4"+C2+"R4 L8"
1450 X(9)=X(1)+X(2)
1460 X(10)=X(3)+X(4)
1470 X(11)=X(5)+X(6)
1480 X(12)=X(7)+X(8)
1490 X(13)=C1+" BR "+S+" BR "+B1+B+" R "+S1+" R|:5 "+R2+" :|"
1500 X(14)="|:4 "+S1+S+"R :|"+C2
1510 /*
1520 for I=0 to 14 : m_trk(1,X(I)) : next
1530 /*
1540 /*         < Bass >
1550 /*
1560 X(0)="     Y49,20 @v127L8 Q8 @1 O3 P3 D4C4>"
1570 X(1)="O2 |:4 G<G> :||:4 F+<F+> :|"
1580 X(2)="|:4 F<F> :||:4 E<E> :|"
1590 X(3)="|:4 D+<D+> :||:4 D<D> :|"
1600 X(4)="|:4 C+<C+> :||:3 D<D> :|EF+"
1610 X(5)=X(1)+X(2)+X(3)
1620 X(6)="|:4 C+<C+> :||:4 D<D> :|"
1630 X(7)="O3 |:4 C<C> :|>|:4 B<B> :|"
1640 X(8)="|:4 G<G> :||:2 E<E> :| G<G>G+<G+>"
1650 X(9)="|:4 A<A> :|<|:4 D<D> :|>"
1660 X(10)="|:4 G<G> :|Q8 O2 G R4.< D4C4"
1670 X(11)=X(5)+X(4)
1680 X(12)=X(5)+X(6)
1690 X(13)=X(7)+X(8)+X(9)+X(10)
1700 X(14)=X(7)+X(8)
1710 X(15)="<|:7 C<C> :| E-<E->>"
1720 X(16)="|:4 G<G> :| G1"
1730 /*
1740 for I=0 to 16 : m_trk(2,X(I)) : next
1750 /*
1760 /*         < Melody >
1770 /*
1780 X(0)="     Y50,00 @V125L8 Q8 @5 O5 P3 D4C4>"
1790 X(1)="|:2 B2&BBB<C> B2&BBB<C>"
1800 X(2)="B2<D2 E1 E-2&E->E-G<E-"
1810 X(3)="D4C4>B4<C4> A2&AC+EA"
1820 X(4)="|1 G4.F+4<DDC> :| G4. F+4. F4"
1830 X(5)="L4 E2<C2> BAGF+ G2<D2 FED>B<"
1840 X(6)="C2G2 F+1 G1 G8R4.DC L8 >"
1850 X(7)=X(1)+X(2)+X(3)+X(4)
1860 X(8)=X(5)+X(6)+X(5)
1870 X(9)="C2G2 F+2G2 G1"
1880 /*
1890 for I=0 to 9 : m_trk(3,X(I)) : next
1900 /*
1910 /*         < Melody Delay >
1920 /*
1930 X(0)="R8  Y51,48 V12  L8 Q8 @5 O5 P3 D4C4>"
1940 /*
1950 for I=0 to 9 : m_trk(4,X(I)) : next
1960 /*
1970 /*         < Bell >
1980 /*
1990 X(0)="     Y52,20 @V118L8 Q8 @3 O5 P3 R4R4 "
2000 X(1)="|:2 @3D@4D @3G@4G @3B@4B <@3D@4D>"
2010 X(2)="@3D@4D   @3F+@4F+ @3A@4A <@3D@4D>"
2020 X(3)="@3D@4D   @3F@4F   @3A@4A <@3D@4D>"
2030 X(4)="@3E@4E   @3G@4G   @3B@4B <@3E@4E>"
2040 X(5)="@3E-@4E- @3G@4G   @3B@4B <@3E-@4E->"
2050 X(6)="@3D@4D   @3G@4G   @3A@4A <@3D@4D>"
2060 X(7)="@3D-@4D- @3E@4E   @3A@4A <@3D-@4D->"
2070 X(8)="@3D@4D   @3F+@4F+<@3D@4D  @3C@4C> :|"
2080 X(9)="O5 @3C<@8C@3G<@8C> @3E<@8C>@3G<@8C>>"
2090 X(10)="@3D<@8D@3A@8F+ <@3D>@8F+@3G@8A>"
2100 X(11)="@3G<@8D@3G<@8D> @3B@8G<@3D>@8B>"
2110 X(12)="@3E<@8E@3B@8A- @3E@8B@3A-@8E>"
2120 X(13)="@3A<@8E@3A@8E <@3C>@8A@3E<@8C>>"
2130 X(14)="@3D<@8D@3A@8F+ <@3D>@8A@3F+<@8D>>"
2140 X(15)="@3G<@8D@3B@8G <@3D>@8B@3G<@8D> P3G @4P3G2..>"
2150 X(16)=X(1)+X(2)+X(3)+X(4)
2160 X(17)=X(5)+X(6)+X(7)+X(8)
2170 X(18)=X(9)+X(10)+X(11)
2180 X(19)=X(12)+X(13)+X(14)
2190 X(20)=X(15)
2200 X(21)=X(18)+X(19)
2210 X(22)=X(15)
2220 /*
2230 for I=0 to 22 : m_trk(5,X(I)) : next
2240 /*
2250 /*         < Brass >
2260 /*
2270 X(0)="     Y53,12 V13  L4 Q8 @6 O5 P3 D4C4>"
2280 X(1)="|:2 B1 B1 B2<D2 E1 C1"
2290 X(2)="C2>B<C> A1 |1 A.A2R8 :| A.A.G+
2300 X(3)="G2<C2 DC>GF+ B2R2 <DC>BA-"
2310 X(4)="A2<C2 D2D2 G1 G8R4.DC>"
2320 X(5)=X(1)+X(2)
```

▶ 7月号のハガキにはミシン目がついていなかった。やっぱりないと不便だということが改めてわかった。 荻原 豊隆(22)長野県

```
2330 X(6)=X(3)+X(4)+X(3)
2340 X(7)="A2<C2 E-2G2 G1"
2350 /*
2360 for I=0 to 7 : m_trk(6,X(I)) : next
2370 /*
2380 /*         < Brass >
2390 /*
2400 X(0)="     Y54,32 V13  L4 Q8 @6 O4 P3 G4G4 "
2410 X(1)="|:2 G1F+1 F2F2 G1 G1"
2420 X(2)="G2GG E1 |1 G.F+2R8 :| G.F+.F
2430 X(3)="E2E2 F+F+F+F+ G2R2 A-A-EE"
2440 X(4)="E2A2 A2A2 B1 B8R4.G4G4"
2450 X(5)=X(1)+X(2)
2460 X(6)=X(3)+X(4)+X(3)
2470 X(7)="R2A2< C2C2> B1"
2480 /*
2490 for I=0 to 7 : m_trk(7,X(I)) : next
2500 /*
2510 /*         < Brass >
2520 /*
2530 X(0)="     Y55,20 V13  L8 Q8 @6 O4 P3 D4D4 "
2540 X(1)="|:2 @6V12 D1D1 D2D2 @7"
2550 X(2)="|1 REGB<E>B P2GE P3"
2560 X(3)="@6 E-1 D2D4D4 @7"
```

```
2570 X(4)="P1 RD-EA<D- P3>A<D-EF+DAD P2GF+ED> P3 :|"
2580 X(5)="|2 <EBGE>B P2GE>B<"
2590 X(6)="@6 P3 E-1 D2D4D4 @7 P1 ED-EA<D-"
2600 X(7)="P3>A<D-EF+DGD P2AD<D>D> @6"
2610 X(8)="L4 P3 C2C2 DDDD L8D2 @V121RGAB L4V13"
2620 X(9)="EERR R2E2 G2G2 G1 G8R.DD L8"
2630 X(10)=X(1)+X(2)+X(3)+X(4)+X(5)
2640 X(11)=X(6)+X(7)+X(8)+X(9)
2650 X(12)=X(8)
2660 X(13)="E2E2 G2G2 G1"
2670 /*
2680 for I=0 to 13 : m_trk(8,X(I)) : next
2690 /*
2700 /*         P e r f o r m a n c e
2710 /*
2720 m_play()
2730 end
2740 /*
2750 /*      Program completed
2760 /*      Sunday Feburuary 25th P.M. 19:05'48"26
2770 /*      From Koji-Tabeta with love.
2780 /*         次も頑張る!
2790 /*
```

リスト4　Artistic Traps

```
1000 /*   P O W E R  D R I F T             S E G A
1010 /*
1020 /*         A r t i s t i c  T r a p s
1030 /*
1040 /*   for Roland U-220        B y  A . K a m o m i y a
1050 /*
1060 /*    Ｕ－２２０のメモリーを書きかえます。実行前に、メ
1070 /*  メモリーの内容をディスクなどにセーブしておいてくださ
1080 /*  い。データ送信の方法は、Ｕ－２２０のマニュアルを参照
1090 /*  してください。ＯＳ上からセーブする場合の操作例を示し
1100 /*  ます（Ver 2 ドキュメントから転載）。
1110 /*
1120 /*   COPY CON MIDI cr
1130 /*   (I) cr                                  （初期化）
1140 /*   (RA) cr                      （ｱｽｷｰﾓｰﾄﾞ録音待機）
1150 /*   ^C                    （CTRLｷｰとCｷｰを同時に押す）
1160 /*   MIDI楽器よりﾃﾞｰﾀ送出      (u220ﾏﾆｭｱﾙ p88 ｻﾝｼｮｳ)
1170 /*   COPY MIDIA A:TEST cr               （ﾌｧｲﾙに落とす）
1180 /*   COPY CON MIDI cr
1190 /*   (R) cr                  （録音ﾓｰﾄﾞ解除／ﾊﾞｯﾌｧｸﾘｱ）
1200 /*   ^C
1210 /*                              詳細はドキュメント参照
1220 /*
1230 m_init()
1240 dim str pd(30)[255],x[255],y[255],z[255]
1250 dim char tim(3,11),set(8),pat(8,12)
1260 dim char po(255)
1270 /*
1280 for i=1 to 9
1290   m_alloc(i,3000)
1300 next
1310 m_sysch("mop")
1320 setting()
1330 pdata()
1340 m_play()
1350 end
1360 /*
1370 /*    S e t  M M L  T o  T r a c k
1380 /*
1390 func trk(t)
1400   r=0
1410   while po(r)<>255
1420     m_trk(t,pd(po(r)))
1430     r=r+1
1440   endwhile
1450   return()
1460 endfunc
1470 /*
1480 /*    B e n d  R o u t i n e  by Zenchan
1490 /*
1500 func str bnd(A;str,L;float,V1;float,V2;float)
1510 str B[256]:int I:float VL,V
1520 VL=(V2-V1)/(L-1):B="":V=V1
1530 for I=1 to L
1540   if V>16383 then V=16383 else if V<0 then V=0
1550   B=B+"@B"+str$(int(V))+A
1560   V=V+VL
1570   if I<>L then B=B+"&"
1580 next
1590 return(B)
1600 endfunc
1610 /*
1620 /*    S e t t i n g  P a r a m e t e r s
1630 /*
1640 func setting()
1650 /*
1660 /*  Timbre Name
          B a s s
1670 tim={   0,     0,     0,     0,     0,     0,     0,     0,     0,
   0,     0,     0,
1680 /*   Tone Media    Dt Vsens Csens Level Atack Decay Sustn
Reles  Fine Coars
1690           49,     0,     5,    15,     8,   127,     8,     8,     8,
   8,    64,    32,
1700 /*  BndLw BndUp ABDep Psens Csens  ABrt VibMd VibDl VibDp
VibRt VibWF VibRT
1710            2,    12,    14,    14,    14,     0,    12,     0,     0,
  50,     0,     0,
1720 /*  VibCS VibPs
          no use
1730            0,     0,     0,     0,     0,     0,     0,     0,     0,
   0,     0,     0}
1740 timbre(1)
1750 /*
     G u i t a r
1760 tim={   0,     0,     0,     0,     0,     0,     0,     0,     0,
   0,     0,     0,
1770           33,     0,     5,    15,     8,   127,     8,     8,     8,
   8,    64,    32,
1780            2,    12,    14,    14,    14,     0,    12,     0,     0,
  50,     0,     0,
1790            0,     0,     0,     0,     0,     0,     0,     0,     0,
   0,     0,     0}
1800 timbre(2)
1810 /*
   B a c k i n g
1820 tim={   0,     0,     0,     0,     0,     0,     0,     0,     0,
   0,     0,     0,
1830          105,     0,     5,    15,     8,   127,     8,     8,     8,
   8,    64,    32,
1840            2,    12,    14,    14,    14,     0,    12,     0,     0,
  50,     0,     0,
1850            0,     0,     0,     0,     0,     0,     0,     0,     0,
   0,     0,     0}
1860 timbre(3)
1870 /*
         M a i n
1880 tim={   0,     0,     0,     0,     0,     0,     0,     0,     0,
   0,     0,     0,
1890          104,     0,     5,    15,     8,   127,     8,     8,     8,
   8,    64,    32,
1900            2,    12,    14,    14,    14,     0,    12,    12,     5,
  55,     0,     8,
1910            0,     0,     0,     0,     0,     0,     0,     0,     0,
   0,     0,     0}
1920 timbre(4)
1930 /*
1940 /*   CrSw  RvSw   Lcd M.tun  RxCh R.Ass R.Chg T.Chg P.Chg
1950 set={   1,     1,    10,   125,    16,     4,     4,     4,     4}
1960 setup()
1970 /*
1980 /*   Patch Name
                   no
1990 pat={   0,   32,    80,   111,   119,   101,   114,    68,   114,
 105,  102,  116,     0,
2000 /*  CrTyp CrOut  CrLv CrTim  CrRt CrDep  CrFb RvTyp RvTim
RvLv  DlFb PrTim    no
2010            0,     0,    24,    14,    20,    10,    32,     3,    20,
  14,     0,     0,     0,
2020 /*  Ctrl1 Prmt1 Ctrl2 Prmt2 Ctrl3 Prmt3 V.Rsv SetUp RcvCh
RxHld RxVol Level  B.Sw
2030            0,     0,     0,     0,     0,     0,     6,     0,    10,
   0,     1,     0,     1,
2040 /* 1  Tim RxVol V.Rsv O.Ass RcvCh Level   Pan KR.Lw RxHld
KR.Hi RxPan VelLv VelTs
2050            0,     1,     4,     1,     1,     0,     0,     0,     1,
 127,     1,     0,     0,
2060 /* 2
2070            0,     1,     4,     1,     2,     0,     0,     0,     1,
 127,     1,     0,     0,
2080 /* 3
2090            0,     1,     4,     2,     3,     0,     0,     0,     1,
 127,     1,     0,     0,
2100 /* 4
2110            0,     1,     4,     2,     4,     0,     0,     0,     1,
 127,     1,     0,     0,
2120 /* 5
2130            0,     1,     4,     2,     5,     0,     0,     0,     1,
 127,     1,     0,     0,
2140 /* 6
2150            0,     1,     4,     1,     6,     0,     0,     0,     1,
 127,     1,     0,     0}
2160 patch()
2170 /*
2180 return():endfunc
2190 /*
2200 /*    U - 2 2 0  Timbre Parameter Setting Function ver 2.
00
2210 /*
2220 func timbre(v;char)
```


▶ "The Master of Payment"の柴田氏は僕と同じようなことを考えていたんだ，と思った。僕も小銭は最小限主義者で，なるべく1円玉や5円玉をもらわないように努力しているのですがあまり徹底できてないのが残念です。　　安尾　文教(23)愛知県

```
 2230 dim int b(14),i,sum=0
 2240 char d,e,w,x,y,z
 2250 /*
 2260 v=v-1 : tim(1,0)=tim(1,0)-1 ·   /* ﾌﾟﾛｸﾞﾗﾑ No. ｼﾌﾄ
 2270 /*
 2280 m_out(&HF0,&H41,&H10,&H2B,&H12,&H2,int(v/2),(v-int(v/2)*2)
*64)
 2290 b(0)=256*tim(0,1)+ tim(0,0)
 2300 b(1)=256*tim(0,3)+ tim(0,2)
 2310 b(2)=256*tim(0,5)+ tim(0,4)
 2320 b(3)=256*tim(0,7)+ tim(0,6)
 2330 b(4)=256*tim(0,9)+ tim(0,8)
 2340 b(5)=256*tim(0,11)+tim(0,10)
 2350 b(6)=tim(1,0)+tim(1,1)*128+tim(1,2)*4096
 2360 b(7)=tim(1,3)+tim(1,4)*16+tim(1,5)*256
 2370 b(8)=tim(1,6)+tim(1,7)*16+tim(1,8)*256+tim(1,9)*4096
 2380 b(9)=tim(1,10)+tim(1,11)*256
 2390 b(10)=tim(2,0)+tim(2,1)*32+tim(2,2)*512
 2400 b(11)=tim(2,3)+tim(2,4)*32+tim(2,5)*1024
 2410 b(12)=tim(2,6)*16+tim(2,7)*256+tim(2,8)*4096
 2420 b(13)=tim(2,9)+tim(2,10)*256
 2430 b(14)=tim(2,11)+tim(3,0)*256+tim(3,1)*4096
 2440 for i=0 to 14
 2450   d=int(b(i)/256)
 2460   e=b(i)-256*int(b(i)/256)
 2470   w=int(d/16)
 2480   x=d-16*int(d/16)
 2490   y=int(e/16)
 2500   z=e-16*int(e/16)
 2510   m_out(z,y,x,w)
 2520   sum=sum+w+x+y+z
 2530 next
 2540 sum=&H80-(sum+2+int(v/2)+(v-int(v/2)*2)*64)and&H7F
 2550 m_out(sum,&HF7)
 2560 /*
 2570 return():endfunc
 2580 /*
 2590 /*    U - 2 2 0   Setup Parameter Setting Function
 2600 /*
 2610 func setup()
 2620 dim int b(3),i,sum=0
 2630 char d,e,w,x,y,z
 2640 /*
 2650 m_out(&HF0,&H41,&H10,&H2B,&H12,0,0,0)
 2660 b(0)=8192*set(0)+4096+set(1)+256*set(2)+set(3)
 2670 b(1)=0
 2680 b(2)=set(4)
 2690 b(3)=4096*set(5)+256*set(6)+16*set(7)+set(8)
 2700 for i=0 to 3
 2710   d=int(b(i)/256)
 2720   e=b(i)-256*int(b(i)/256)
 2730   w=int(d/16)
 2740   x=d-16*int(d/16)
 2750   y=int(e/16)
 2760   z=e-16*int(e/16)
 2770   m_out(z,y,x,w)
 2780   sum=sum+w+x+y+z
 2790 next
 2800 sum=&H80-(sum)and&H7F
 2810 m_out(sum,&HF7)
 2820 return():endfunc
 2830 /*
 2840 /*    U - 2 2 0   Patch Parameter Setting Function
 2850 /*
 2860 func patch()
 2870 dim int b(38),i,sum=0
 2880 char d,e,w,x,y,z
 2890 /*
 2900 for i=3 to 8                      /*MIDI ch ｼﾌﾄ
 2910   pat(i,4)=pat(i,4)-1
 2920 next : pat(2,8)=pat(2,8)-1
 2930 /*
 2940 m_out(&HF0,&H41,&H10,&H2B,&H12,0,&H6,0)
 2950 b(0)=256*pat(0,1) +pat(0,0)
 2960 b(1)=256*pat(0,3) +pat(0,2)
 2970 b(2)=256*pat(0,5) +pat(0,4)
 2980 b(3)=256*pat(0,7) +pat(0,6)
 2990 b(4)=256*pat(0,9 )+pat(0,8)
 3000 b(5)=256*pat(0,11)+pat(0,10)
 3010 b(6)=2048*pat(1,5)+64*pat(1,2)+pat(1,4)
 3020 b(7)=8192*pat(1,0)+128*pat(1,6)+pat(1,8)
 3030 b(8)=2048*pat(1,10)+pat(1,3)
 3040 b(9)=32768*pat(1,1)+4096*pat(1,7)+64*pat(1,11)+pat(1,9)
 3050 b(10)=256*pat(2,1)+pat(2,0)
 3060 b(11)=256*pat(2,3)+pat(2,2)
 3070 b(12)=256*pat(2,5)+pat(2,4)
 3080 b(13)=16384*pat(2,10)+8192*pat(2,9)+256*pat(2,8)+32*pat(2,
7)+pat(2,6)
 3090 b(14)=128*pat(2,12)+pat(2,11)
 3100 for i=0 to 5  /*ﾊﾟｰﾄ 1 - 6
 3110 b(i*4+15)=8192*pat(3+i,3)+256*pat(3+i,2)+128*pat(3+i,1)+pa
t(3+i,0)
 3120 b(i*4+16)=4096*pat(3+i,6)+32*pat(3+i,5)+pat(3+i,4)
 3130 b(i*4+17)=32768*pat(3+i,10)+256*pat(3+i,9)+128*pat(3+i,8)+
pat(3+i,7)
 3140 b(i*4+18)=256*pat(3+i,12)+pat(3+i,11)
 3150 next
 3160 for i=0 to 38
 3170   d=int(b(i)/256)
 3180   e=b(i)-256*int(b(i)/256)
 3190   w=int(d/16)
 3200   x=d-16*int(d/16)
 3210   y=int(e/16)
 3220   z=e-16*int(e/16)
 3230   m_out(z,y,x,w)
 3240   sum=sum+w+x+y+z
 3250 next
 3260 sum=&H80-(sum+&H6)and&H7F
 3270 m_out(sum,&HF7)
 3280 /*
 3290 return():endfunc
 3300 /*
 3310 /*    P l a y   D a t a
 3320 /*
 3330 func pdata()
 3340 m_tempo(167)
 3350 /*
 3360 /*    ドラム
 3370 /*
 3380 pd(0)="@n10 o2 @v120 @u120 l8 r4 <c32c32c16>f8a32a32a16>b8
< @b8192[do]
 3390      x="r>b<d4>bb<d>brb<d4>rb<d>brb<d4>bb<d
 3400 pd(1)=x+">b<r<c32c16.c32c16.>>r64b16..<a32a16.a32a16.r64'f
16..>b<'>b<
 3410 pd(2)=x+">bb<@u80'df'@u100'df'@u120'df'd>b<d>b<
 3420 pd(3)=x+">b<l12<cccc>aaaffl8>bb<
 3430 pd(4)=x+"r<c32c16.>@u80'da'@u100'df'@u120'df'r>brb<
 3440 pd(5)="|:2r>b<d4>bb<d>b<:|r>b<d4>bb<d4>bb<dd&ddd>b<
 3450      x="r>b<d4>bb<d4>bb<d4>bb<d>bb4<d>br4<d4>b4<d>b<r
 3460 pd(6)=x+">b<d>b<
 3470 pd(7)=x+"<c>a>b<
 3480 pd(8)="|:7r>b<d4>bb<d>b<:|<c>>b<a>bbb<af
 3490 pd(9)="r>b<d4>bb<d>brb<d4r>b<d>brb<d4>bb<d4>b<d'df''df'rl1
6@u60d@u80d@u100d@u120dl8>b<
 3500      x="r>b<d4r>b<d>brb<d4r>b<'df'>brb<d4r>b<
 3510 pd(10)=x+"l16@u80d@u100d@u120ddl8>bb<d>brb<f>b<
 3520 pd(11)=x+"d>c<rl16<@b10924c32c32ccc@b8192c32c32ccc>aaaaff
 3530 pd(12)="ffffl8d4r>b<d>brb<d4r>b<'df'>brb<d4r>b<d<c16>a16>b
b<d>brb<f>b<
 3540 pd(13)=x+"d>brb<d4r>b<'df'>brb<@u40d@u60d@u80d@u100d@u120d
dddddr>brb< [loop]
 3550 po={0,1,2,3,4,5,6,7,8,1,9,6,7,8,10,11,12,13,255}
 3560 trk(1)
 3570 /*
 3580 /*    ベース
 3590 /*
 3600 pd(0)="@n1 @m0 @b0,127 @b8192 @p64 @1 o1 q7 @v120 @u110 l8
 r4 r4.g& [do]
 3610 pd(2)="|:23g:|<d&dd>a+<c>ga+<c>g&
 3620 pd(3)="|:23g:|<c>fffffg<c>g&
 3630 pd(4)="|:23g:|<c>fga+g&ggrf&
 3640 pd(5)="|:28f:|rff+g&
 3650 pd(6)="|:16g:||:15f:|g&|:16g:||:15f:|a&
 3660      x="aaaaaaag&gggggg
 3670 pd(7)=x+"a&"+x+"a&
 3680 pd(8)=x+"b&|:11b:|<crd&d>g&
 3690 pd(9)="|:24g:|fga+g&g4.g&
 3700 pd(10)=x+"b&|:11b:|<crd&de&
 3710      x="eeeeeeed&dddddddc&
 3720 pd(11)=x+"cccccccc>b<crcrdre&
 3730 pd(12)=x+"|:13c:|dre&
 3740 pd(13)=x+"|:28c:|rcr>g& [loop]
 3750 po={0,2,3,2,4,5,6,7,8,2,9,6,7,10,11,12,11,13,255}
 3760 trk(2)
 3770 /*
 3780 /*    ディストーション  ギター
 3790 /*
 3800 pd(0)="@n2 @m0 @b0,127 @b8192 @p54 @2 o3 q2 @v100 @u110 l8
 r4 r4.g& [do]
 3810      x="g<q8g>q2gg<q8gg>q2g
 3820 pd(1)=x+"gg<q8g>q2ggg<q8g>q2g<q8g>q2
 3830 pd(2)=x+"rr<q8g>r<fff>r<g>q2
 3840 pd(3)=x+"r<q8rfrffga+>q2g&
 3850 pd(4)=x+"r<q8fga+g4grf&
 3860 pd(5)="f>q2ff<q8 f4>q2ffffff<q8f>q2f<q8f4.>q2fff<q8f>q2fff
ffff<q8frff+g&
 3870      x=" g1&g2.>q2gg<q8f>q2ff<q8f>q2ffff<q8f>q2ff<q8f>q2fff
<q8
 3880 pd(6)=x+"g&
 3890 pd(7)=x+"a&
 3900 pd(8)="a2..g>q2 ggg<q8 g>q2ggg<q8a&
 3910 pd(9)="a2..g>q2ggg<q8g>q2ggg<q8b&b1bbr<crd4>q2>@v120g&
 3920 pd(10)="g<q8g>q2gg<q8gg>q2ggfff<q8g4.rg&
 3930 pd(11)="a2..g>q2ggg<q8g>q2ggg<q8b&b1bbr<crd4e&
 3940      x="e2..d1c&c1>b<crcrdre1d1c1&c2&c&
 3950 pd(12)=x+"cdre&
 3960 pd(13)=x+"c1&c1rcr>@v110>q2g& [loop]
 3970 po={0,1,2,1,3,1,2,1,4,5,6,7,8,8,9,1,2,1,10,6,7,8,8,11,12,1
3,255}
 3980 trk(3)
 3990 /*
 4000 /*    バッキング
 4010 /*
 4020 pd(0)="@n3 @m0 @b0,127 @b8192 @p26 @3 o5 q8 @v80 @u110 l8
r4 r2 [do]
 4030      x="drrer4.f&fferderdr4.er4.fr2r
 4040 pd(1)=x+"g4.
 4050 pd(2)=x+"e4.
 4060 pd(3)=x+"rr>f&
 4070 pd(4)="@v60f1a1<c1f2.r@v80d&
 4080 pd(5)="d1&d2.r4frrfr2frrfr4.d& d1&d2.r4crrcr2crrcr4.e&
 4090 pd(6)="e2..dr4.dr4.e&
 4100 pd(7)="e2..dr4.dr4.e&e1d+&d+4erf+4r
 4110 pd(8)=x+"r4d&
 4120 pd(9)="e2..dr4.dr4.e&e1d+&d+4erf+4g&
 4130     x ="g2..f+&f+2..e&e1dererf+rg&g2..f+&f+2..e&e2..&e&e2&
 4140 pd(10)=x+"ef+rg&
 4150 pd(11)=x+"e1&e1rerr [loop]
 4160 po={0,1,2,1,3,4,5,6,6,7,1,8,5,6,6,9,10,11,255}
 4170 trk(4)
 4180 /*
 4190 /*    バッキング
 4200 /*
 4210 pd(0)="@n4 @m0 @b0,127 @b8192 @p63 @3 o5 q8 @v80 @u110 l8
r4 r2 [do]
 4220      x=">a+rr<cr4.d&ddcr>a+<cr>a+<r4.cr4.dr2r
 4230 pd(1)=x+"d4.
 4240 pd(2)=x+"c4.
 4250 pd(3)=x+"rr>c&
 4260 pd(4)="@v60c1f1a1<c2.r>@v80b&
 4270 pd(5)="b1&b2.r4<crrcr2crrcrrr>b& b1&b2.r4arrar2arrarrr<c&
 4280 pd(6)="c2..>br4.br4.<c&
```

▶ 我が家のキーボード（1991年7月号参照）は，その後耐水ペーパーで磨かれて元のつや消しに戻りましたとさ。やっぱりX68000はつや消しが一番っすね。

澤田　裕史(15)神奈川県

```
4290 pd(7)="c2..>br4.br4.b&b1b&b4<crd4r
4300 pd(8)=x+"r4>b&
4310 pd(9)="c2..>br4.br4.b&b1b&b4<crd4e&
4320      x="e2..d&d2..c&c1>b<crcrdre1d1c1&c2&c&
4330 pd(10)=x+"cdre&
4340 pd(11)=x+"c1&c1rcrr [loop]
4350 trk(5)
4360 /*
4370 /*      バッキング
4380 /*
4390 pd(0)="@n5 @m0 @b0,127 @b8192 @p85 @3 o4 q8 @v80 @u110 18
r4 r2 [do]
4400      x="frrgr4.g&gggrfgrfr4.gr4.gr2r
4410 pd(1)=x+"a+4.
4420 pd(2)=x+"g4.
4430 pd(3)=x+"rr>a&
4440 pd(4)="@v60a1<c1f1a2.r@v80g&
4450 pd(5)="g1&g2.r4arrar2arrarrrg& g1&g2.r4frrfr2frrfrrra&
4460 pd(6)="a2..grrrgrrra&
4470 pd(7)="a2..gr4.gr4.f+&f+1f+&f+4gra4r
4480 pd(8)=x+"r4g&
4490 pd(9)="a2..gr4.gr4.f+&f+1f+&f+4gra4b&
4500      x="b2..a1g&g1f+grgrarb1a1g1&g2&g&
4510 pd(10)=x+"garb&
4520 pd(11)=x+"g1&g1rgrr [loop]
4530 trk(6)
4540 /*
4550 /*      シンバル
4560 /*
4570 pd(0)="@n10 @u110 o3 14 r4 r4.c+8 & [do]
4580 pd(1)="c+>@u100|:11a+:|rr2r8<@u110c+8&
4590 pd(2)="c+>@u100|:11a+:|rr2<@u110c+8c+8&
4600 pd(3)="c+>@u100|:11a+:|rr.<@u110c+>@u100a+8&
4610 pd(4)="a+ |:13a+:|18r<@u110c+c+>@u80f+
4620 pd(5)="|:8@u100f+@u80f+:||:2@u80f+f+@u100f+a+@u80f+f+@u100
f+@u80f+:|
4630 pd(6)="|:8@u100f+@u80f+:|@u80f+f+@u100f+a+@u80f+f+@u100f+@
u80f+f+f+@u100f+a+@u80f+f+@u100f+<@u110c+>
4640 pd(7)="r@u80f+|:27<@u120d+>@u80f+:| rf+r<@u100c+r@u110c+rc
+&14
4650 pd(8)="c+>@u100|:12a+:|r218r@u80f+
4660 pd(9)=" c+>@u100|:10a+:|rf+f+f+f+8<@u110c+8&
4670 pd(10)="c+>@u100|:11a+:|f+f+f+f+
4680 pd(11)="f+<@u110c+>@u100|:10a+:|f+f+f+f+8<@u110c+8&
4690 pd(12)="c+>@u100|:14a+:|18a+a+r1r2..<@u110c+& [loop]
4700 po={0,1,2,2,3,4,5,6,7,1,8,5,6,7,9,10,11,12,255}
4710 trk(7)
4720 /*
4730 /*      メロディ
4740 /*
4750 pd(0)="@n6 @m0 @b0,127 @b8192 @p63 @4 o4 q8 @v122 @u125 18
r4 r2 [do]
4760 pd(1)="|:19r1:| r2rq4dq8g<d&
4770      x=bnd("e",4,8192,8875)
4780      y=bnd("c",16,8192,6826)
4790 pd(6)="d1&d2rer164"+x+"&e2&e8.@b819218efc2&164"+y+">@b8192
18q4dq8g<d&
4800      x=bnd("b",4,8192,8875)
4810 pd(7)="d1&d2rer>164"+x+"&18b2&b.@b8192brf2.ab<c&
4820      x=bnd("c",8,8192,9668)
4830 pd(8)="c2>b2e4.grar<c&c2>b4.<164"+x+"&18c@b8192dcr4>ab<c&
4840 pd(9)="c2>b2e4egrgaf+&f+1d+&d+4erf+4g&
4850      x=bnd("g",16,8192,7509)
4860      y=bnd("g",16,7509,6826)
4870 pd(10)="g4&164"+x+"&
4880 pd(11)=y+"|:6r1:|r2..18@b8192q4dq8g<d&
4890       x=bnd("d",8,8192,8875)
4900 pd(13)="c2>b2q6e4.g4a4q8b&b1<164"+x+"&d418@b8192erf+rg&
4910       x=bnd("a+",4,8192,8875)
4920 pd(14)="g2.ba&a2f+2f+grc&c1rf+rg&g2.164"+x+"&a+1618@b8192a
&a2<d4.c1&c1>g&
4930       y=bnd("f+",8,8192,8875)
4940       z=bnd("a+",4,8192,8875)
4950 pd(15)="g2.164"+x+"&a+1618@b8192a&a2f+2164"+y+"&f+418@b819
2c4&c1f+rg&g2.164"+z+"&a+1618@b8192
4960       x=bnd("b",8,8192,8875)
4970 pd(16)="a&a2<d4.e1&e1&e1&e2&er>164"+x+"18@b8192bg [loop]
4980 po={0,1,6,7,8,9,10,11,6,7,8,13,14,15,16,255}
4990 trk(8)
5000 /*
5010 /*      ユニゾン
5020 /*
5030 pd(0)="@n6 @m0 @b0,127 @b8192 @p63 @4 o3 q8 @v120 @u90 18
r4 r2 [do]
5040 trk(9)
5050 /*
5060 return()
5070 endfunc
```

(善) のゲームミュージックでバビンチョ

皆さん，コナミの「生中継68」のBGMは聴きましたか？　ありゃ凄いですよ。曲調はいまハヤリの「T-SQUARE」に代表されるジャパニーズフュージョン！　シンセブラスで奏でられる美しくもキャッチーなメロディ。曲も凄いけど音もいい。ボコーダーボイスのサンプリングにも結構びっくりだけど，なんといってもFMで出しているギターの音がお見事。ディストーションからフランジがかったコーラスサウンドまでをFM音原数声で表現しているんです。ぜひ，ご一聴あれ！

●出たな!! ツインビー　CD:KICA-7503
キングレコード　2,800円(税込)

「やってくれたな，コナミめ」というのが私の感想。いままでのコナミ節を生かしつつも，ちょっと映画音楽のような壮大な雰囲気のフレーズやフィルを盛り込んだりして，思わず「うーむ」。ひさびさに胸が熱くなりましたよ，ほんと。内容は，アレンジ4曲とオリジナルサウンド。まあ，構成は月並みなんですが，アレンジの出来がこれまた最高ですな。私は「サンダークロス」のアレンジで感動したクチなんですが，アレを超えるデキです。エンディングのアレンジは，ほとんど坂本龍一のノリですが，これもなかなか。とにかく，最初から最後まで本当に楽しい1枚です。

・SEアラカルトの「ガッチンコ」の「ガ」が聞こえない。しかもこれが女の子の声で叫ばれるもんだから……。

お勧め度　10

●天使たちの午後・初恋編　CD:APCG-4012
アポロン　2,600円(税込)

Hなノリを目指したのか，それともアニメとかのイメージアルバム的なものを目指したのか，イマイチどっちつかずの中途半端な内容。声優さんの音程がまったく外れているのに，OKサインを出したプロデューサーはいったい……（スタッフにちゃんとコカコーラ飲ましたか)。音楽のほうももうちょっとなんとかしてほしかった。「打ち込み」なのは責めません，音の厚みをなんとかしてよね（レソラシのイントロは誤解を招くぞ)。

・「みつめてなんとか」のほうが面白かった。次回作に期待。

お勧め度　5

●パーフェクトセレクション パロディウスだ！
CD:KICA-1032 キングレコード 3,000円(税込)

「パロディウスだ！」の曲は私は大好きだ。X68000版もテープに録音して毎日聴いてる(バランステストの「カエルの歌が」がなかなか良かったりする)。で，そんな私はこのCDを聴いての感想はまず「おいおい」だった。内容は全曲同ゲームのBGMのアレンジ曲なのだが，あんな素晴らしい題材を与えられてよくもまあこんなヘンチクリンなアレンジをしましたねってな感じ。M1やSY77のプリセットもろもろなのは許すとしても，編曲がなってないから聴いていて疲れる。アレ？　でもコナミのアレンジバージョンって，いつも最高峰のデキじゃなかったっけ？　……コナミのサウンドスタッフがノータッチなのかね。

・ライナーノーツで山下章氏もいっているがX68000版のサウンドを収めたCDも出してほしいな。

お勧め度　5

●ローリングサンダー2　CD:VICL-15005
ビクター音産　1,500円(税別)

トランペットのバック，サックスのメロディ，アコースティックバスにオルガンのコーディング＆ソロ。ブラシにライドシンバルの響き。んんー，ジャズともロックとも区別のつけられないこの曲調，ノリは'70～'80年代のアメリカの刑事もののドラマのBGM。メロディアスなゲームミュージックというわけではないが，とても聴きやすくていい。作曲者はかなりプロの匂いがする。FM音源で鳴っているサックスとブラスはとってもいい音なんだけど，せっかくサンプリングがあるんだからサックスだけでもPCMでやってほしかった。

・値段が1,500円ってのはいいよね

お勧め度　7

●765（NAMCO）メガミックス　CD:APCG-4014
アポロン　2,000円(税別)

ラリーXから妖怪道中記までの，ナムコのオールドタイトルから中オールドタイトルまでのBGMを，ハウスやディスコのノンストップのノリでアレンジしたもの。懐かしのフレーズに懐かしの効果音を盛り込んだりしてるのでなかなか面白いんだけど，1回聴けばいいかなという感じ。少なくとも普段ダンスやラップを聴いてる人は，アリガチという印象を受けるはず。が，いままで普通のゲームミュージックを聴いていた人にとっては結構インパクトがあるかもしれない。しかし，ナムコサウンドというのはいまにしろ昔にしろそれ単体で完成されたものだからね，アレンジは難しいんだろうね（突然知ったようなことをいう（善))。

・「おろおろおろおろおろかもの」には1時間笑い続けましたよ。

お勧め度　6

＊

今月はちょっと評価が厳しかったかな。ま，私個人の意見だから，みんな参考程度に留めよう，過信はよくないかも。「西川さん『天午後』のCD最高じゃないっすかぁ」というはがきは，真っ向からうけてたつぞ。かかってきなさい。いやーん。

CG OSAKA'91
COMPUTER GRAPHICS
イメージテック'91

日本ビクターのイメージWS

IRISでフライトシミュレータ

6月12日（水）から15日（土）の4日間，マイドームおおさか，大阪商工会議所ビル，大阪コクサイホテルで，第7回を迎える「COMPUTER GRAPHICS OSAKA '91」（略称，CG OSAKA '91）が開催された。

展示会では70社の企業が多彩なグラフィック機能を搭載したエンジニアリング・ワークステーションや，各種のグラフィック入出力機器，プレゼンテーションシステムなどのCG関連ハード，ソフトを展示。プロフェッショナル用途のCG機器ならではの高機能を見せつけていた。また，自社の製品を使用してのCGアニメーションデモなども当然ながら前面に出ていて，目を楽しませてくれた。

展示会とは別に，シンポジウム，特別企画も催され，人工現実感（つまり，バーチャルリアリティ）体験コーナーや，ゴルフシミュレーション体験コーナーが設置されていたが，整理券をもらって順番を待たなければ参加できないほどの大盛況となっていた。

高画質，ピクトログラフィ

昇華型フルカラー，「PhotoMaker」

あちこちでこんなデモが見られた

神鋼電機のCDS

恥ずかしそうに人工現実感体験

会場での「μイメージシンセサイザー」デモ

縮小画像によるイメージディレクトリ機能

プレゼンテーションでの使用例（色変換）

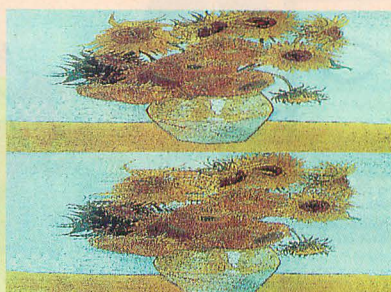
μイメージシンセサイザー

CG OSAKA'91にシャープから出展されていた「μイメージシンセサイザー」はX68000，およびその周辺機器によって構成された業務用画像処理システムである。

用途は建築，景観，インテリアをはじめとする各種デザインシミュレーションということだが，音声画像データベースとしての機能も充実しているので，さまざまなプレゼンテーションにも活用できる。

編集機能はあたり線を描くガイドプレーンによる正確な位置合わせでの2画面合成，画像に対しての多彩なエフェクト，7つのエリアを登録でき，わかりやすく色分け表示されるマスク機能など，各種デザインやシミュレーションの検討に適した多くの機能を用意。

出力装置は通常のカラープリンタやビデオプリンタのほかに，ピクトログラフィやPIXEL DiOなども選択できるので，業務用途に耐えうる高画質プリントが可能である。価格はシステムの構成により異なるが，約230万円から。

マスク機能を使って床をフローリングにしてみる

現在のところはないが，2D画像をもとに3Dスコープ用の画像を作るといった機能も研究中

[第3回]

脳のなかのイメージを視覚化する

寺尾 響子

小学生のころ，夏休みには必ず海へ行きました。湘南の逗子に親戚の家があって，行きたいときに行き，好きなだけ泊まったのです。そのころの湘南はごくあたりまえの海で，今みたいに，ウエストコースト風白いテラスレストランもなければ，ホットドッグプレス系茶髪サーファー少年もいませんでした。家からスクール水着のまま，すたすたとゴム草履で海まで歩いて15分ぐらい。午前中おもいっきり泳いで，お昼ごはんを食べに，またすたすたと戻ります。昼下がり。おとなは疲れて昼寝をするけれど，こどもの私は寝るのがなんだかもったいない。「外に出るなら帽子をかぶるのよ。日射病になるから」という母の声も無視して，飛び出します。少し湿った土のにおいのする真夏の道はうんと暑く，空気はどんよりと停止し，とても静かです。人は皆休んでいるのでしょう。ただ，蟬の声だけが耳に痛い。

毎年めぐってくる夏の湿ったにおいをかぐと，記憶がよみがえってきます。記憶というよりも，私自身をひたしていた空間，いっさいのものを含めてといったほうがよいかもしれません。それが今回のCGになりました。

響子inCGわ〜るど

触れようとしても　触れられぬ

原風景

静かに坐って

ただ　眺めるためだけのもの

イメージからCGへ

イメージを絵にするにはどうしたらいいのでしょう。これが絶対に正しいという方法はないと思います。人にはさまざまな生き方があるように，その表現のしかたも人それぞれだから。ここではどうやって，私が脳のなかのイメージを視覚化してCGにしたかを，簡単にお話しするつもりです。[1]

まず，頭をからっぽにします。連想します。「真夏の昼下がり」「海」，連想される色は青です。それもにごりのない，透明な濃いブルー。コバルトブルー。夏の空の色。作品の基調となる色にします。「暑い」「のどが渇く」，氷にしようか水にしようか迷いました。メタボール[2]を試してみたかったので水に決定。女の子のキャラクターは登場させるにしても，今までと同じでは芸がありません。こぼれる水の不安定な感じを出すために，やじろべえにしました。「日射病」「外に出るなら帽子を」，それで帽子をかぶせました。麦わら帽子にしようかとも思いましたが，色がベージュだと夏の明るい感じが出ないので，黄色の水玉に変更。おおまかなデザインを頭のなかでしてしまいます。

もういちど作品を構成する要素を1つひとつ吟味し，自分の脳のなかのイメージと照らし合わせて，目に見える形にとり出してゆきます。水の入ったグラスのときは，まずラフスケッチを手で描きました。グラスのイメージをより明確にするために，「透明でポップな感じ80%」「あっ，こぼれ

た！　どうしよう……心配度25%」などと言葉を書きそえたりもしました。そして……。

X68000のまえに座ります。CGの入力作業に移ります。グラスは円錐と楕円体と円柱を組み合わせてつくりました。これらの物体の大きさ，位置，色，屈折率，透明度，表面反射率などの属性を数値を入れ替えながら，気に入る画像が得られるまで計算しなおします。

今までの一連の作業は，プログラムの実行のように逐次的に行っているわけではありません。私の脳はきわめてファジィでニューロ[3]，あいまいなので，気ままに試行錯誤を繰り返しながら，同時並行で進めていきます。CGの作品づくりでいちばん楽しいのがこの作業です。思いついたイメージをどんな色や形にしようかと考える時間。感覚脳と論理脳のはざまを，振り子のようにゆっくりと，行きつ戻りつするときです。

このCGもこんなふうにしてつくりました。でも，なんだか少し違うのです。たしかに，あの暑さあのにおいあの空間は，くっきりと脳に刻み込まれているのに，そのものではないのです。文章で書いても違う，絵で描いても……違う。いつもそうでした。つくっているときは楽しい，でも出来上ってしまうとちょっとずれている感じ。手をのばしてつかもうとすると，するりと心の暗がりへ逃げ込んでしまう。きっと，脳のなかのイメージが原形であるならば，表現というフィルタを通して生成したものはあくまでも写像にしかならないのでしょう。それでも，原形に近づくために，私はこれからもせっせと作品をつくり続けるにちがいありません。

1）本当は絵解きはしたくありません。作品を見てそのまま感じとってほしいのです。
2）3次元空間に2つ以上の点を配置し，各点のまわりに生ずる濃度に応じて曲面を生成させるアルゴリズム。もともとは，大阪大学を中心に開発されたもの。
3）厳密な意味で使っているわけではありません。CMに出てくる家電製品を形容しているのと同じと思ってください。念のために。
4）写真は1280×1024ドットのフルカラー（1670万色）画像を4×5インチのポジフィルムに出力したものです。物体数は206（うちメタボール数11），光源は2です。

SX-WINDOW対応グラフィックツール

Easypaint SX-68K

Takahashi Tetushi 高橋 哲史

待望のSX-WINDOW用アプリケーションとして登場したのが，このEasypaint-68Kです。Easyといっても決して低機能というわけではありません。では，マ○ジン奨励賞で勢いづく高橋先生(笑)に紹介していただきましょう。

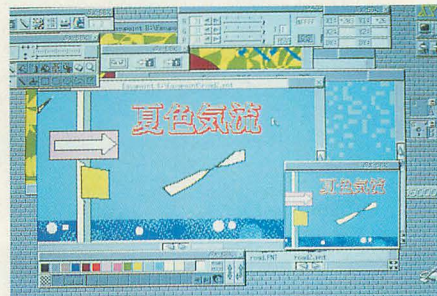
さてさて，X68000にグラフィックツールは数あれど，マルチウィンドウでお絵描きができるのはこのEasypaintだけなのです(高橋注：本当はいくつかあるのですが故意に無視してます。だってどれもタコなんだもん)。だからついつい意味もなく何枚もウィンドウを開いて前衛芸術の花を咲かせてしまったりするお茶目な私です(笑)。数枚の絵に囲まれながらも後ろでちゃっかり「信州」が動いてるとこなんかも実にSX心をそそります。ちなみについ先日数十回のトライの末やっと信州終わりました。エンディング画面にドラゴンは出てこなかったけれど，なかなか素敵な画面で感動してしまいました(うるうる)。ということでこれはEasypaintの紹介です。

Easypaintは大きく分けて3つのプログラムで構成されています。まず，本体のEasypaint.X，スキャナとの仲を取り持つEasyscan.X，そして描いた絵の印刷を行うEasyprint.Xです。これらのプログラムはそれぞれに独立しており，単体で呼び出してもきちんと機能します。またプログラムを組める人であるならば，自作のプログラムからこれらを呼び出して，プリンタ，スキャナを利用することが可能になっています。う～んこの辺はさすがですね。それでは早速Easypaint.Xから使っていってみましょう。

これがEasypaintだっ

皆さんご承知のようにSX-WINDOWで扱えるグラフィックは768×512モードで色数は65536色中の16色と相場が決まっていますから（もちろん例外もありますが)，Easypaintもそれに合わせて設計されています。よって使用可能色は16色，絵のサイズは1×1～1024×1024（ウィンドウごとに可変）となっています。それとSX-WINDOW上で動いていますので，当然メモリを喰います（こういう言い方はちょっと語弊があるかな？)。もっともメインメモリが2Mバイトあればそんなに不自由することはありません。今回のレビューもほとんど私のマシン（X68000 ACE-HD,2M）で行っていますし。しかし，さすがにそろそろメモリを増設したくなっちゃいましたね（でもお金がないの……)。

というわけで，お絵描きに取りかかりましょう。メニューウィンドウからツール選択ウィンドウを呼び出します。

このツール選択ウィンドウはなかなかよく出来ていて，必要な描画モードをすぐに選択できるように工夫されています。具体的にどのようになっているかといいますとアイコン上段にペン・ブラシ・スプレーなどの「道具」が配置され，その下段に点・直線・塗り潰しなどその道具の「使い方」が並んでいるのです。これにより，「え～とCIRCLEFILLは『バケツ』+『円』だな」といった直感的な選択が可能となっています。これはなかなか賢い設計といえるでしょう。

SXらしくさまざまなウィンドウと共存するEasypaint

ペンギン……もといペン先編集の様子

ルーペ（2倍）で細かい部分を修正

なにはともあれ，絵を描くにはペンとブラシが必要ですね。Easypaintのペン・ブラシはなかなか凝っていて，ペン先の形状定義はもちろんホットスポットの設定までできるようになっています。ホットスポットというのは要するに，「マウスカーソルの先端部分が実際のペン先のどこを指しているか」を表すものです。ホットスポットが無条件にペン先の真ん中になっているとペンの形状によっては「あれ～？　こんなとこまで描くつもりはなかったのにぃ～」といった事故にたびたびあったりしますが，自分のフィーリングに合うようにこれを設定しておけば気持ちよくさくさく描画できるわけです。

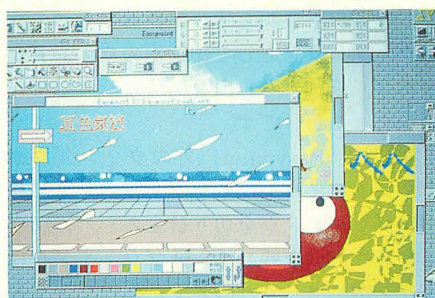
また，ペン先の大きさがX軸Y軸それぞれの方向に1ドット単位で変えられるというのもなかなか面白いです。たまに自分で

は思いもよらぬ形のペン先が出来たりしてなかなかときめきがあります(笑)。あと使えそうな機能として「縁取り」が挙げられます。これはその名のとおり縁取りをするのですが，これまでのグラフィックツールではお見受けしなかった機能だけに新鮮でした。

そのほか一般的な直線・曲線・楕円・閉曲線もきちんと用意されています。あ，それから「面とり長方形」といったお初の用語にもお目にかかりましたが，これは要するに「角の丸まったBOX」ということです(最初はなんのことかと思い結構あせりました)。

お次はスプレーです。ここでもホットスポットの設定に加えてスプレースピードの調節ができるようになっていたりして，なかなかの気配りを感じさせますが，機能自体は従来のソフトと大差ありません。またバケツ（一般的なグラフィックツールでは「ペイント」にあたります）も過不足なく機能しています。文字描画にしても消しゴムにしてもごく普通ですね。

さて，これらの描画機能を使う際には，必ずパレットウィンドウのお世話になります。ここではカラーとタイルパターンの設定を行います。カラーの設定にはHSV値，RGB値の両方が使用でき使いやすくなっています（私はどうも頭がRGBしてるのでHSVのみのツールだとちょっと困ってしまうのです)。またタイルパターンも32種類登録でき，ひとつのタイルパターンに2種類のカラーが割り当てられるようになっています。と，ここで気がつくのですがこのパレットウィンドウにはグラデーション関係の設定が何もありません。そう，残念ながら「お客さん，Easypaintではグラデーションは扱っていないんですよ，ええ」なのです。16色といえどタイルパターンを駆使すればグラデは可能なのでぜひとも取り入れてほしかったのですが……。

次は「ハサミ」。いわゆるカット＆ペーストです。カット範囲の指定も「ペイント指定」,「閉曲線指定」など多彩で「選択範囲の逆転」などと使い合わせると，かなり緻密に範囲指定ができ，複数ウィンドウの絵の合成なども手軽に行えるようになっています（ただし，当然ですがこの辺の機能は思いっきりメモリを喰います)。またコピー範囲の内容はクリップボードに送られますので，イメージデータを取り扱うことのできるほかのアプリケーション（もちろん，SX-WINDOW上のもの）とのデータ交換手段としても利用できます。

もうすっかり夏ですね～。あ～海行きたいな～

またマウス座標表示サブウィンドウもしっかり用意されていますので，精密な描写も可能です。環境設定ウィンドウでアンドゥの有無，描画起点の変更（左上or中心）も選べるようになっています。

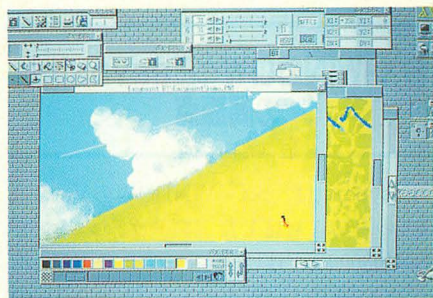
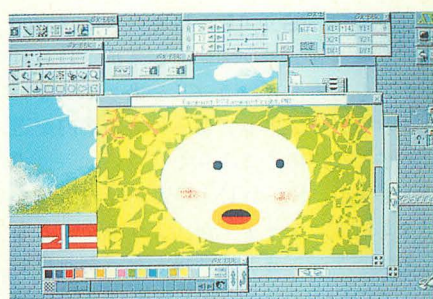
Easyscan&Easyprint

さてEasyscan，スキャナです。福原さんのように「マウスで直接なんでも描けるぜ！」といった超巧絶技を持つ人を除けば，やはりスキャニングは必須の手段といえましょう。マニュアルに「スキャニングには4Mバイト以上のメモリが必要です」と一瞬くらっとするような文章があったりしますが，それはEasypaintと同時にEasyscanを呼び出したときのお話で，直接Easyscanする場合は2Mあれば十分足りますのでまずはご安心を。

まず，Easyscanは2回に分けてスキャニングを完了します。まずプレスキャンを実行してスキャナでの取り込み可能範囲をすべて高速に粗くスキャンし，そのあとにそこからカラースキャン範囲を選択して実際の絵がスキャニングされるわけです。カラースキャンにはさまざまな設定があり，微調整のきく取り込みが可能となっています。

そして最後は「印刷」です。当然モノクロ/カラーの両方で印刷できます。モノクロ印刷にはSX-WINDOWがもともと持っているシステムを利用しています。環境設定も当然そのままなので違和感なく利用できると思います。またカラー印刷では上記の設定に加えてカラーアジャスト設定も用意されていますので画面上のイメージをおおかた遜色なくプリントアウトできるでしょう。

試用後の感想

絵を描く機能についてはほぼ申し分なく揃っていると思います。欲をいえば，色の変換機能がほしかったところです（私はこの機能を結構多用するもので)。あとルーペ

ブラシを多用して描いてみました

SXFACE(笑)。背景は閉曲線ペイントで簡単に

内で使えるのが，「点」と「消しゴム」のみというのも辛いですね。せめて「直線」と「ペイント」くらいは出来るようにしてもらいたかったものです。それとやや特殊効果が弱いでしょうか？　「モザイク」や「ぼかし」なんかがあるとなにかと便利なのですが(「16色でぼかすなんて無理だよぉなどといった声が聞こえてきそうですが，他機種のグラフィックツールにはできるものが実際にあります)。

それとマスク関係がないのもちょっと残念でした。あとパレットウィンドウ，ルーペウィンドウ，ツール選択ウィンドウなどの優先順位が固定されてしまっているのもZ'sSTAFFに慣れた体には少し酷というものでしょう。

しかし，キーボードショートカットが充実しているので，慣れてくるとかなりスピーディな処理が可能になるのではとも感じました。またたいていの描画機能にSHIFTキー，CTRLキーによるモードモディファイアがついているのも評価できます（直前の座標から描画を再開したりといったことが簡単にできる)。

ということでこのEasypaint，ペインティングツールとしてはまずまずの出来栄えではないでしょうか？　全体的に手軽なグラフィックツールというコンセプトながら，結構細かいところで多機能な面も目につきました。ただ，SX-WINDOW対応のアプリケーションとしてみれば，細かい機能を省いてもウィンドウ環境を生かす工夫のほうにもう少し力を入れたほうがよかったのではと思います。

SX-WINDOWとEasypaintの問題点

Easypaintは，SX-WINDOWのVer.1.1以上のものに対応したアプリケーションです。X68000 XVI以外のユーザーの方は，市販のSX-WINDOW Ver.1.1（9,800円）のパッケージを購入する必要があります。ただし，すでにVer.1.0をお持ちの人は，4,800円でバージョンアップサービスが受けられます。もちろん登録カードを出していないと，バージョンアップサービスの案内が送られてこないのでご注意ください。

ともあれ，SX-WINDOWが発表されて早1年以上になりますが，ようやくまともなアプリケーションの登場となったわけです。このEasypaintにもまだまだ課題は残されていますが，やはりウィンドウ同士でデータのやりとりができるというのは素晴らしいことですね。ハサミ（カット&ペースト）なんかを使ってると「わははは，おいらは今ウィンドウ上で絵を描いてるんだぜいっ。ぶらぼーっ！」と実感させてくれます。あっちのウィンドウからこっちのウィンドウに，はたまたその逆をと自由自在に切り貼りができるのはいい気持ちです。

それからほかのアプリケーションを実行しつつ絵が描けるというのも強いですね。そのうちSX-WINDOWでもレイトレのソフトが出て，Easypaintの後ろで計算させといて，画像が出来上がったらすぐにEasypaintでお料理なんてこともできるかも（わくわく）。

ただし手放しで喜んではいられない部分もいくつかあります。ここでは，ウィンドウ上のアプリケーションとしてEasypaintを見た場合の問題点を考えてみましょう。

サブウィンドウをなんとかして

まず，Easypaintのウィンドウの扱いについてです。Easypaintには，メニューウィンドウと，そこから呼び出される6つのサブウィンドウ，そして作図用のウィンドウがあり，それらを自由な場所に配置して作図ができるわけです。

問題は，Easypaint以外のウィンドウやディスクアイコンなどをクリックしてアクティブウィンドウを切り替えると，作図ウィンドウだけを残して，サブウィンドウは次々と閉じてしまうのです。作図ウィンドウをアクティブにすれば，サブウィンドウは再びオープンするわけですが，ほんのちょっとファイル操作をしたり，別のウィンドウを参照したりしたいだけなのに，わざわざ何枚ものウィンドウが次々と閉じたり開いたりするのを眺めるのはたまりません。メニューウィンドウに一括してサブウィンドウを閉じるボタンを用意するとかして，自分が閉じたいときに閉じるというのが正しいやり方ではないでしょうか。

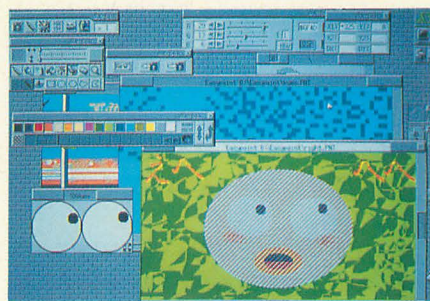
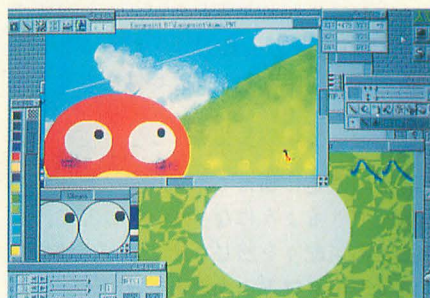
システム上の都合かなとも思いましたが，ピンボール.Xなどはうまくサブウィンドウを扱っている例でしょう。どうしてもできないというのであれば，いっそサブウィンドウをひとつにまとめるべきかもしれません。マルチウィンドウであることをあだにしてはもともこもありませんからね。

SX-WINDOW ver.1.1　9,800円（税別）
Easypaint SX-68K　12,800円（税別）
X68000用（要2Mバイト）

肝心のコピー機能は？

さて，先ほどからいっているように，ウィンドウ環境で絵を描く最大のメリットは，絵と絵の間で自由にカット&ペーストが使えるということです。その意味から見ると，Easypaintのコピー機能には重要な機能が抜けています。簡単なことですが，絵の上に別の絵を合成するためには透明色の指定がなんとしても必要なのです（透明度の指定まではEasyなpaintツールということで望みません）。もっともEasypaintには，連続するカラー領域を選択できるという便利な機能もあるのですが，ちょっと複雑なパターンになるとうまくいきません。

16色パレットの呪い

これはSX-WINDOW自体の問題でEasypaintにとやかく言ってもしょうがないのですが，SX-WINDOWのグラフィックのパレットは，65536色中の「任意の」16色と決まっています。これはかなりのくせ者です。なぜならばSX-WINDOWはその名のとおりウィンドウシステムだからです。いっぱいのウィンドウが開いてそれぞれの仕事をしています。その中には何枚かのグラフィックもあるでしょう。しかし一度に使える色は16色なのでパレットは強制的にアクティブウィンドウのそれに統一されます。必然的のほかのグラフィックは不幸に表示されることになるでしょう。

顔全体を塗り潰しでカット範囲指定したところ

他のウィンドウにペースト。色が変わってしまう

SX-WINDOWが出たばかりのころは，グラフィックも眺めるだけで，まあそんなものかなと思っていたのですが，実際に絵を描いて，ウィンドウ間のデータのやりとりをしようとすると問題の大きさを実感させられます。せっかくいろいろな絵に流用できそうな部品をライブラリにしておいても，パレットが違う絵には使えません（無理矢理やるとさらに不幸になりますし）。

その辺はEasypaintも苦慮しているようでパレットのコピー機能を用意したり（複数ウィンドウでパレットを合わせて描く場合に使う），「なるべくカラースキャンのパレット値に合わせて絵を描きましょう」とマニュアルに提唱してあったりもします。

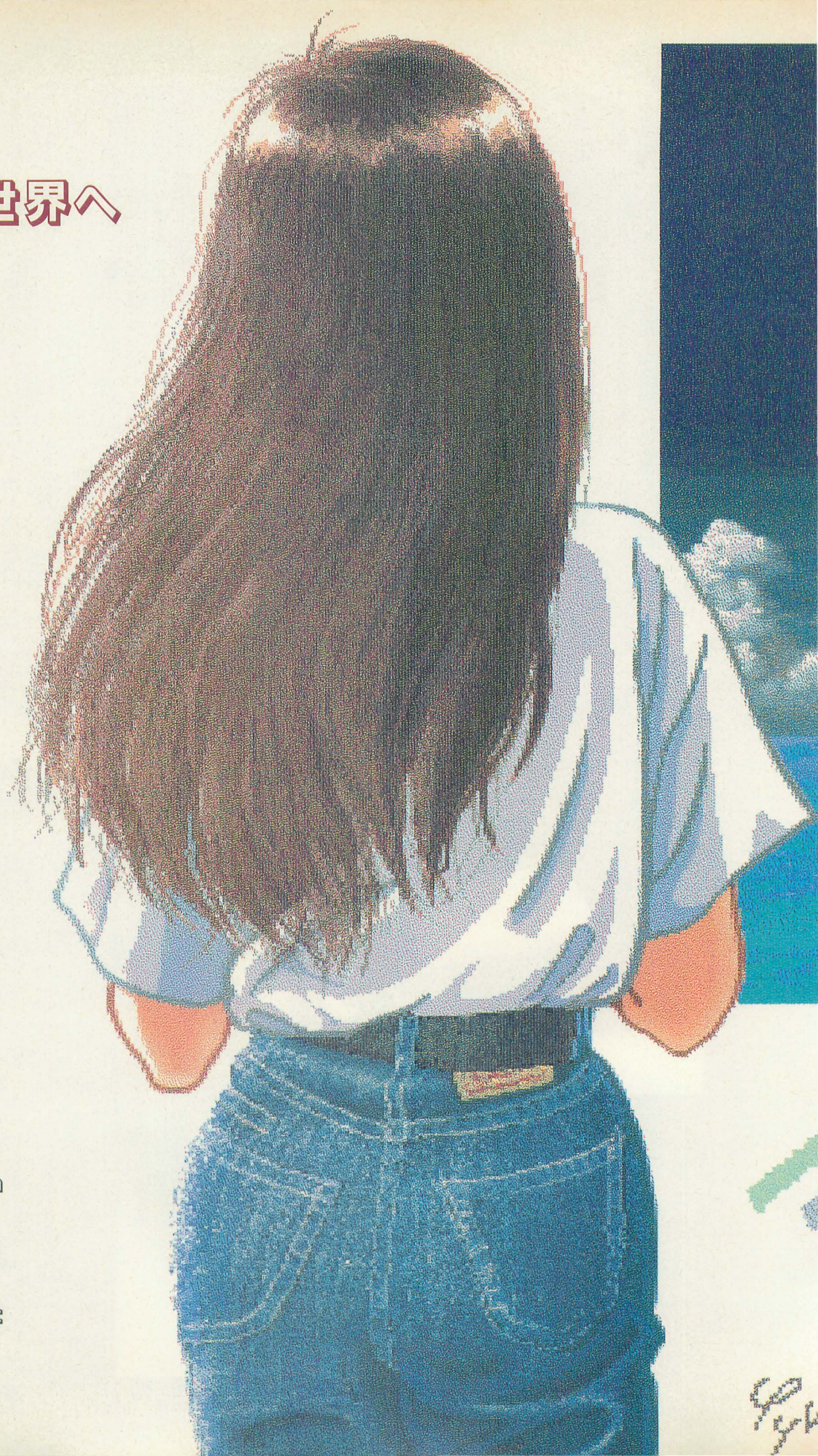
ところで，前回の付録ディスクに収録されていたSXIMAGEでは，この問題を「どんな絵でも統一されたパレットに合わせて表示する」ことで解決しています。そりゃ，私だって「せっかく65536色あるのに……」とついぐちってしまいます。65536色の中から選べるとなればいろんな色を使ってみたくなるのが人情というものですからね。でも，そこはやはりウィンドウであることのメリットを生かすためにも標準パレットというものを検討してみるべきかもしれません。ということで，よい解決法がみつかるまではユーザーがパレットを上手に管理して資源を有効活用するしかなさそうですね。ああ，16色の呪縛。

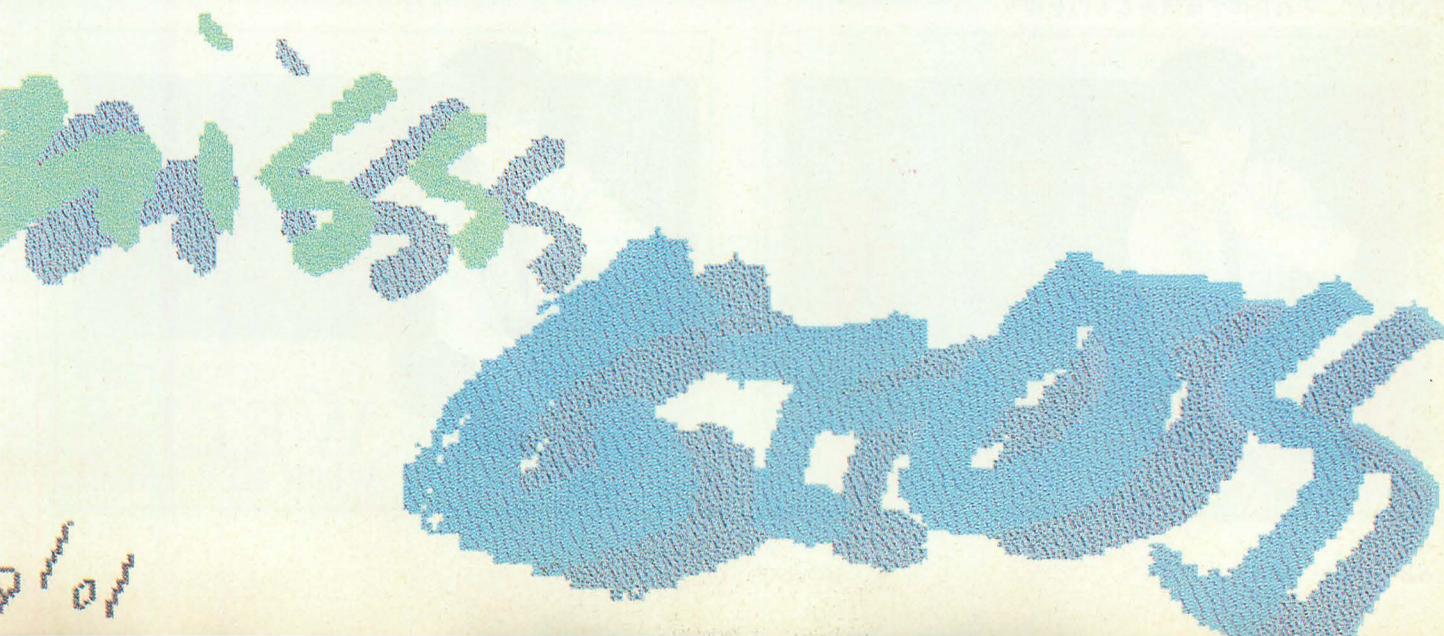
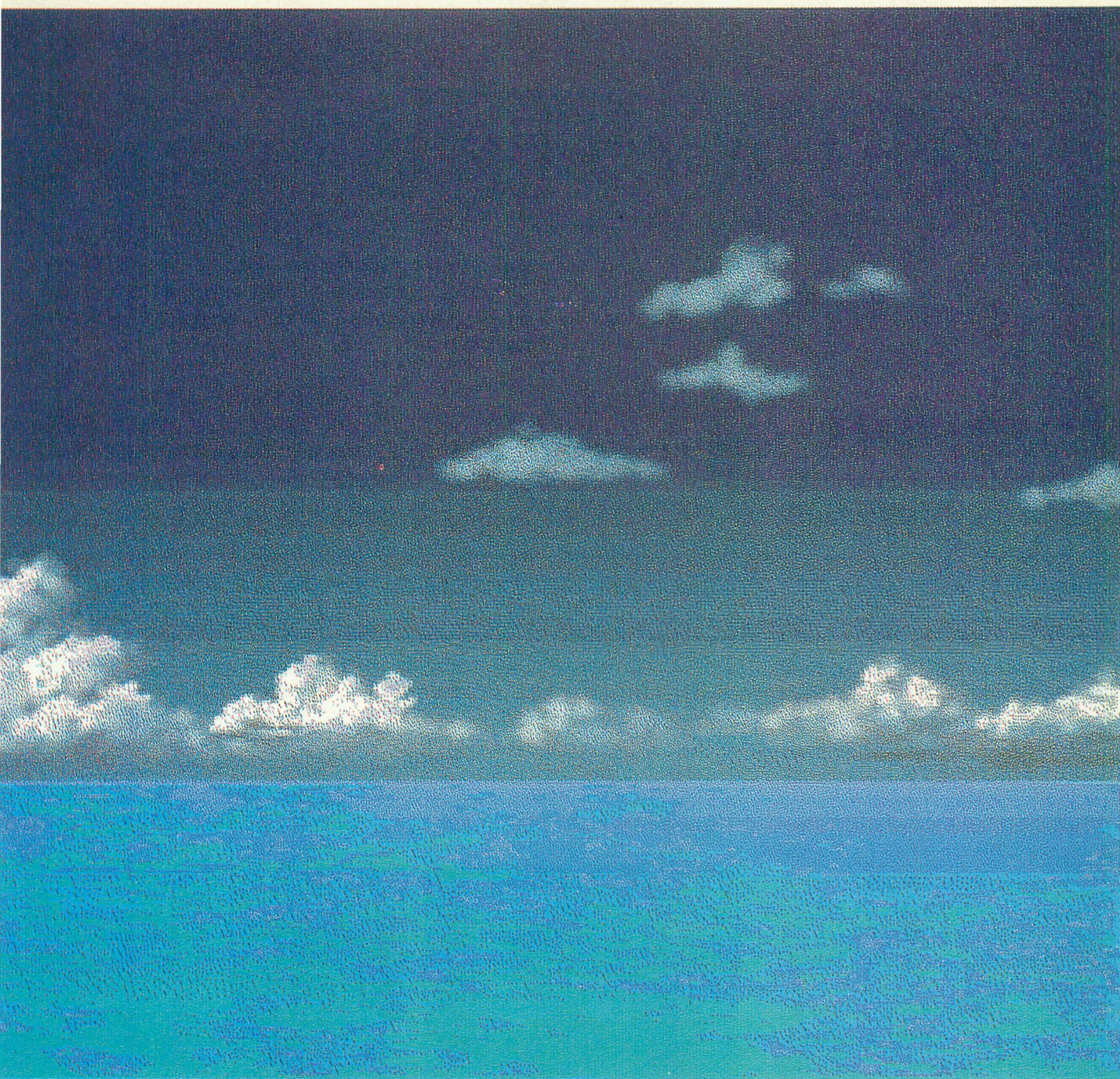
特集

印刷の世界へ

出力装置としてのプリンタ。その2つの流れは高解像度化とカラー化です。フルカラーグラフィックをどのように表現するか？ それに対する解答のひとつを私たちは知っています。ですから，あとはそれをどのように扱っていくか？　それを考えればよいわけです。
特にSX-WINDOWなどに見られる，アプリケーション実行環境のビジュアル化はテキストデータとグラフィックデータの融合を促進することでしょう。より人に優しいインタフェイスとして，WYSWIG化は必然的な流れを持って私たちのもとにやってきます。
画面で見えるものをそのまま編集し，そのとおりのものを出力する。この当たり前のようなことを実現するために多くの人が努力しています。画面のままを出すなら簡単？　いえ，むしろ最高の品質のものを出力しつつ，それをどのように画面で扱うか，が問われるのです。前提になる「最高の品質の出力」。私たちはここから確保していかなければなりません。
あなたのプリンタは十分に使われていますか？

CONTENTS

特集
印刷の世界へ

"MISS GEOS"
by
Tohru Fukuhara

"OUTPUT"
by
IO-735X-B
Color Image Jet

プリンタを使うということ

出力デバイスを探る

Nakano Shuichi 中野 修一

まずはプリンタの基本概念から。でも，せっかくのカラーページですから，浜崎氏のハードコピーの出力を織り交ぜました。あなたもお手持ちのIO-735Xでこんなに綺麗なハードコピーが取れるのです。

それは情報に形を与える

プリンタ。アンケートによれば，読者の所有率でダントツの周辺機器です。特にCZ-8PC3/4/5などの24/48ピンカラー熱転写プリンタが多数を占めています。

いったいなんに使っているのかというと，大半の人は日本語ワープロを使うためと答えるでしょう。個人レベルでは（プログラム開発を主体にしている人は別にして）応用範囲の広い熱転写プリンタが人気を集めていることからも，それはうかがえます。

PRINTER……

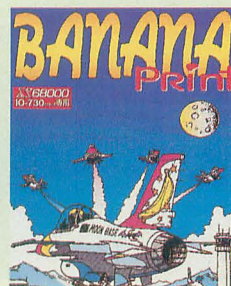
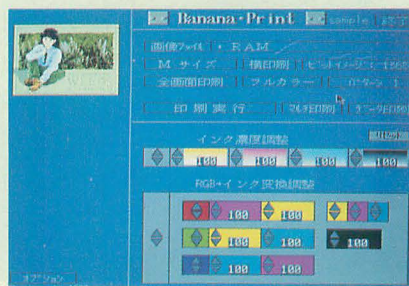
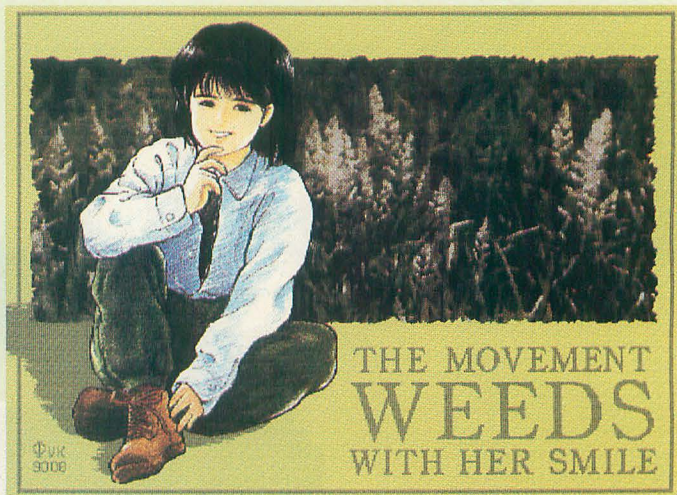
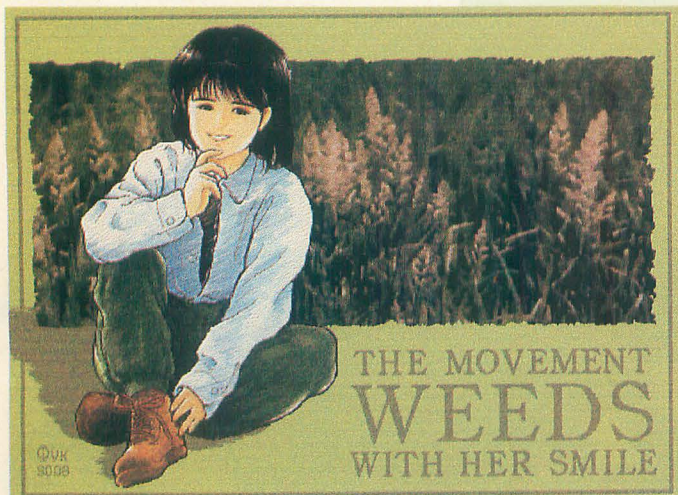
それは「印刷屋」をも表す単語です。各自の作成した文書ファイルを打ち出したり，65536色を駆使したカラーグラフィックのハードコピーを取ったり……，とかく情報（多くの場合，符号化された電気信号）だけのやり取りに終始するコンピュータの世界において，プリンタは情報を私たちの手にできるかたちに出力してくれる数少ない機器のひとつです。

ときとして，プリンタの出力したものはコンピュータを使った作業の最終目的物となります。単なる情報の出口としての機能のほかに，それ自体で価値あるものを作り出せる装置なのです。

プリンタあれこれ

まずは人気の熱転写方式（リボンについたインクを熱で溶かして印刷する方式）です。最近の熱転写プリンタは印字速度もかなり速くなっていますし，夜中でも使用できる静粛性，手頃な値段と高い解像度（48ピンの場合），カラー出力などの魅力があります。ランニングコストの高さも感熱紙やマルチタイムリボンを使うことで抑制することができますし，ここ一番というときの印字品質はピカイチかもしれません（個人的には48ピンのフォントは細すぎてあまり好きではない。エプソンの24ドットフォントが最高だと思う）。

Oh!X編集室を見るかぎりドットインパクトがまだまだ主流で一部インクジェットが使用されている程度です。大量のリスト出力や文書出力にはドットインパクトが有利であることは明白です。ドットインパクトプリンタはインクの染み込んだリボンを紙に叩きつけることにより印刷します。頑丈でランニングコストも低く，メンテナンスも不要です。最大の欠点はやはり騒音でしょうか。

ほかの編集部をのぞいてみると，ドットインパクトタイプは少なくほとんどレーザープリンタに置き換えられています。レーザープリンタはコピー機のドラムにレーザーを当てることで電荷を与え，トナー（カーボンの黒い粉）を吸着/転写していくというものです。とにかく高速で1ページ単位

左はZ'sSTAFF，右はBANANA PRINTのIO-735Xでの出力例（46%）。BANANAはディザ法，Z'sは誤差拡散か？　色合いはずいぶん違う。BANANAでは分割出力の超大型プリントも可能だ。
BANANA PRINT 48,000円（税別）ムーンベース
☎022(271)9700

に編集が行え，解像度と値段が高いことが特徴です。

それほどリストを打ち出さないから（打ち出すのはOh!XとOh!FMくらいのものか）かもしれませんが，一説によると昔，某C社（キヤノンではない）がレーザープリンタ市場に参入した際に各編集部に1台ずつレーザープリンタを置いていったのだそうで，そのことも影響しているかもしれません。当時，高価な周辺機器の代名詞だったレーザープリンタをポンと置いていったのですからたいしたものです。これが「Oh!Xを除くすべての編集部に」ではなかったらC社の評価もさぞ上がったことでしょう。

さて，X68000にもレーザープリンタを接続することは可能ですが，その場合はたいていPC-PRエミュレーションモードで使用することになります。こういった場合，テキストファイルの出力は高解像で行われますが，WP.Xなどの出力は綺麗になりません。X68000でレーザープリンタ対応のプログラムというのはTeXくらいしかありませんので，プリンタ本来の機能を使用するためにはプログラムを自作する必要があります（Multiwordは対応するらしい）。

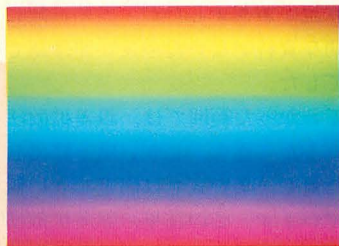
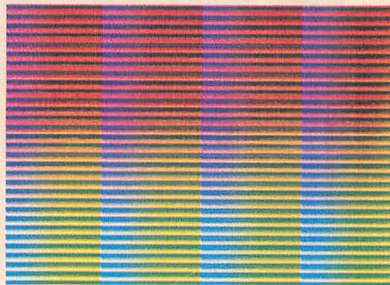
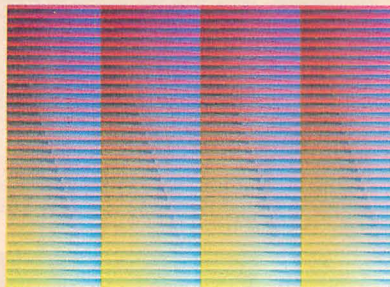
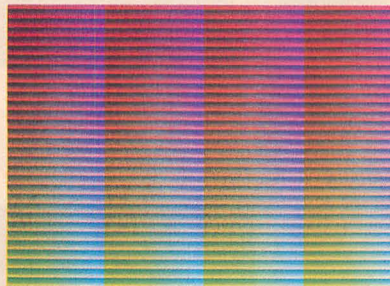
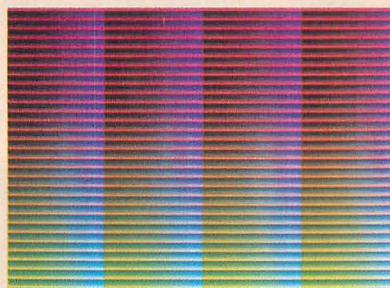
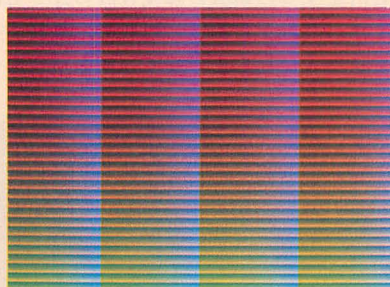
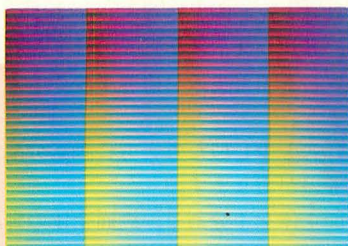
パーソナルで使うレーザープリンタにはまだ決定版というものがないようですが，今後の方向性を考えるとやはり避けては通れません。

カラー出力

X68000でのプリンタ出力を考える際にはカラー画像出力も忘れることはできません。X68000用には熱転写，ドットインパクト，インクジェット，熱昇華型の4種類の方式でカラープリンタが用意されています。というよりも現在ラインアップされている純正プリンタは，CZ-8PK10以外すべてカラープリンタとなっています。グラフィック機能の強力さを反映してか，相当に徹底したカラー重視戦略といえるでしょう。

さて，プリンタでグラフィックを扱う際には必ずといってよいほど，量子化（要するに階調落とし）が必要になります。実際，ビデオプリンタ以外のカラープリンタではドットごとに8色の指定しかできません。

量子化の手法として，Oh!Xではいわゆる「桒野式」が主流で，多くの記事で扱われています。現状では計算コストと品質から最良の方法といっていいでしょう。

出力結果が類似していることから，これまで桒野式と誤差拡散法は同じものではな

各方式による色再現の違い。各方式とも暗部の青の発色が弱く，明るい緑は強すぎる（インクの特性か？）。色変換版は色相をまたいだグラデーションに対応しきれておらず，変換の傷跡が生々しい。また熱転写は独特の発色をすることがわかる。

いかという説がありましたが，誤差拡散に関する適切な資料がないため長い間確認できませんでした。

これまでわかった範囲では，誤差拡散法が周辺ドットから逐次，しきい値を算出していくのに対し，桒野式では固定しきい値を採用しています。この結果，誤差拡散法は画面の局所単位で精度が高く，桒野式では画面全体での輝度誤差がきわめて低いという特徴が考えられます。しきい値の算定以外はほとんど同じ処理といってよいでしょう。計算コストは桒野式のほうが軽いのでわざわざ誤差拡散を研究することもないでしょう。

桒野式と少し似た考え方に画素分配法というものがあります。

画面全体を4分割し，画面全体の輝度の合計を4つの領域の輝度合計の比で分配します。これを各領域について，最終的に1ピクセルになるまで繰り返すというものです。この方法も画面全体での輝度誤差がほとんどありません。再帰的に定義すればプログラムは簡単ですが計算量は膨大となります。逆順に計算してテーブルを作っておけば実用になるでしょうか。なかなか面白いアルゴリズムです。

＊　＊　＊

ということで，グラフィックツールのエフェクト関係や画像解析などを本格的に行いたいという人には『画像解析ハンドブック』（東大出版会）という本がおすすめです。25,750円という定価もべらぼうですが，各種処理のアルゴリズムを一堂に集めており内容はきわめて充実しています。この本のおかげで桒野式の位置づけもやっと明確になってきたのです。

Z's STAFF

48ピン熱転写

BANANA PRINTA

24ピン熱転写

桒野式（色変換なし）

ビデオプリンタ

桒野式（色変換つき）

実際にグラフィックを出力して比較。濃い青から緑へのグラデーション，薄い緑など，絵の色づかいはインクジェットの苦手な配色となっている。Z'sはデフォルトでは白い色に弱い。48ピンの熱転写プリンタは性能が発揮しきれていないことがわかる。

ビデオプリンタとは

X68000（およびX1）のプリンタラインアップのなかにひとつだけ異質なものが加わっています。それはビデオプリンタCZ-8VP1です。ビデオプリンタは普通のプリンタとはかなり違った位置を占めるプリンタです。少なくともこれでアセンブラのソースリストを打ち出す人はどこにもいないでしょう。

ビデオプリンタはビデオ信号およびアナログRGB信号（ただし15kHz）を直接入力してプリントアウトすることが可能です。しかし，この状態ではB6サイズの用紙の1/2くらいが余白になってしまううえ，画面全体を収めることはできません。

このビデオプリンタには画像入力用の端子だけでなく，コンピュータコントロール端子が設けられています。すなわち，通常のプリンタケーブルで接続して，印字を制御できるのです（通常のコード印字はできません）。

ビデオプリンタを直接コンピュータコントロールして画像を出力するソフトとしては1989年9月号で掲載したvprint.xがほぼ唯一のものでしょう。画質は見てのとおり，色再現はいまひとつのところもありますが総合的な画質では他を寄せつけません。この方式でも画面の端は欠けますが，RGB入力時に比べればわずかなものです。

ただし，ビデオプリンタは専用紙と専用フィルムを使うことからランニングコストは他方式とは比べものにならないほどかかります。1枚あたり50円くらいでしょうか。ハガキ用紙もありますから，用途を割り切って使えば，大きさに制限があることはさほど問題にはならないでしょう。

プリンタの使い方

どのプリンタを使ってもプリンタである以上，一定の規則に従って動作します。またどんなことができるか，というのもそれほど変わりません。ここではプリンタを制御する際の手順を追ってみましょう。

まず，ケーブルで接続します。当たり前のことですが，方式を問わずほとんどのプリンタが同一のケーブルによって接続可能です。ただしエプソン製のプリンタはシャープ純正のケーブルではうまく制御できません。メーカーに確認してください。

いったん接続してしまえば，ほとんどのプリンタでコード印字が可能です。そもそもプリンタというものは，ケーブルから文字コードを受け取って，それを印字するという機械なのです。たとえば，

A>COPY CONFIG.SYS PRN

とプリンタを表す“PRN”にファイルをコピーしてやると，プリンタにはCONFIG.SYSの内容が印字されるはずです。これはプリンタドライバになにを使っているかにはほとんど影響されません。漢字が化けたり，改行幅が大きくなっている可能性はありますが，基本的な半角文字はたいてい出力されるはずです。

文字コードを受け取って文字を出すだけの機械でどうしていろんなことができるのでしょうか。およそプリンタと名のつくものは，通常は使わないコードに特殊な意味を持たせておき，その文字が現れるとあらかじめ決めていた仕事を実行するように作られています。これが制御コード（コントロールコード）と呼ばれるものです。

制御コードのうち，ESCコード（キャラクタコード1B$_H$）は特に重要な意味を持っています。たいていESCという文字のあとには，込み入った一連のコードがつながっており，何文字かでひとつのコマンドに対応しているのです。これらの一連のものをまとめてエスケープシーケンスと呼びます。実際にどのようなコードを送ればどんな動作をするかというのはプリンタのマニュアルに詳しく書かれています。

ま，ここらあたりの制御方法はディスプレイに文字を表示する場合とほとんど同じです。ディスプレイ（コンソール）にはそれ用の制御コード，プリンタには機種ごとに違ったコードが設定されています。

プリンタに文字表示以外の動作をさせようとすると制御コードを調べてプリンタに転送しなければなりません。グラフィックを印刷したいとか，プリンタが持っている別の書体を使用したいとか，横幅何文字の設定で打ち出したいとかいうことはなにもグラフィックツールやワープロソフトを使わなければできないことではありません。

実際にプログラムからプリンタを扱うにはファイル操作の知識が必要です。X-BASICでの制御については浜崎氏の記事を見てもらえればわかるように，“LPT”というファイルへ制御コードを書き込むことで行います。先ほどの“PRN”と同様にプリンタを表すファイル名です。

“PRN”は文字データ専用
“LPT”は主に制御データ用

のように使い分けることになっています。

ファイルにコードを書き込むこと，書き込んだコードが命令として動作することを理解すれば誰にでも自由にプリンタを使いこなすことができるのです。

色変換の実際。BANANA PRINT（左）との比較。一長一短があるものの，色変換によって，元の写真との極端な違いはなくなっている（と思う）。それでも誤差部分やまだ不整合な部分はいくつか見られる。まだまだ改善できるはずだ。

そしてDTPへ

やはりプリンタ活用のひとつの終着点はDTP（デスクトップパブリッシング）でしょう。欧米ではもはや一般的になりつつあり，個人での文書作成と実際のパブリッシングがかなり近いところに位置しています。しかし，日本では話題が先行しているだけ

で，たまにMacintoshやワークステーションを使った本作りがされている程度，パソコンユーザーでなければポータブルワープロの付加機能の一種くらいにしかとらえていない人もいるのではないでしょうか。

理由としては，ソフトウェアで欧米での実情にあわせた製品作りがされており，それが日本の実情とはそぐわないということもさることながら，まだそのようなものは必要とされていないという事実が大きく作用しているようです。

本来あるべきパーソナルなDTPというのは，たとえば，角の魚屋さんがパソコンに向かって，ふんふんと特売のチラシをデザインするような，そう，1本の極太マジックに代わるべきものなのでしょう。しかし現時点で，400dpiのアウトラインフォントで出力されたPTA会合のお知らせがどう受け入れられるのかには疑問があります。

かといって本格的な「出版」ということに対してはまだまだ力不足です。そのひとつには日本語の書体が少ないことが挙げられるでしょう。

デスクトップでないパブリッシング，たとえば，Oh!Xの本文では写研の12Q MM-OKLという写真植字の書体が使用されています。これは「12級ミディアム石井明朝オールド仮名ラージ」の意で，12級という大きさ（1級＝0.25mm）の中くらいの太さの石井明朝という明朝体で書体は旧版，仮名は大きく，という指定を満たすものです。

文字の太さをゴナという書体（ゴシック・ナール）で見てみると，

ゴナL　清く正しく美しく
ゴナM　清く正しく美しく
ゴナD　清く正しく美しく
ゴナDB　清く正しく美しく
ゴナB　清く正しく美しく
ゴナE　清く正しく美しく
ゴナU　清く正しく美しく

となります。文字の太さだけでなく，

ナール
シャンゼリゼ大通り
スーシャ
明朗会計3000円ポッキリ
淡古印
新装開店大出血サービス
ナカフリー
あなたは神を信じますか?
イクール
かいしんのいちげき
ゴカール＋ゴナ
この紋所が目に入らぬか！

といった書体の豊富さや，さらには，

長③
システムエラー1が発生しました
平③
ファイルが見つかりません
右長斜③
このファイルは複数実行できません

オリジナルTシャツを作る

昔，Oh!MZ時代には「プリンタごっこ」と称してプリントごっこを使用した多色刷りのカラー印刷システムがなんとプリントごっこの改造例つきで掲載されたことがありました。

同じことをやるのもなんなので，代わりに今回みつけてきたのが「T-boy」です。これはプリントごっこの豪華版といったところで，比較的ちゃんとしたシルクスクリーン印刷が可能なものです。

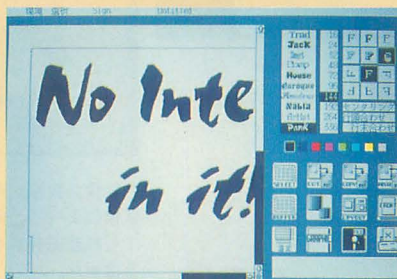
ただ，当初予想したよりも解像度が悪く（16ピンビットイメージが限界か？），正確な多色製版が難しかったので，グラフィックの多色分解はあきらめました。フルカラーのTシャツというのも捨てがたかったのですが，ここではわりと普通のオリジナルTシャツを作ってみました。名づけてX68000（元祖型）Tシャツです。

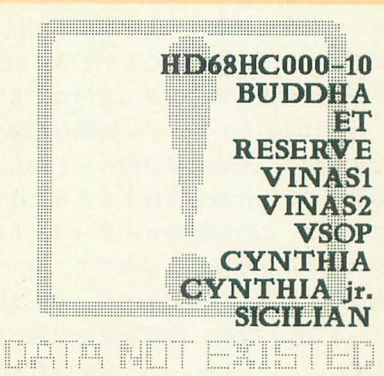
さて，このような作業にパソコンを使うメリットはなんでしょうか？　特に有利なのは文字の表示です。試行錯誤のたびにインレタを使ったり，レタリングしたりしなくてすみますから。ここではアウトラインフォントを使って文字を主体とした図案を作りました。データはすべてNEW PrintShop PRO-68K ver.1.0で作成しています（忍耐の人と呼んでください）。

コツとしては，

1）　できるだけ薄い紙に出力する
2）　できるだけ濃く出力する
3）　規定時間よりやや短めに焼き付ける

といったところでしょうか。

実際にTシャツにするとT-boyではワンポイント（14×14cm）しか作れません。もっとちゃんとしたものを作りたいときは，ひとクラス上の「Tシャツくん」をおすすめします。

T-boy	10,800円
Tシャツくん	29,000円

太陽精機(株)ホリゾン事業部　☎0422(48)5119

といったさまざまな指定が可能です。

実際の“ノン・デスクトップパブリッシング”で使用されているものと比べてみると，日本語ワープロの「豊富な文字装飾」がいかにも馬鹿げたものだということがわかります。

欧文ならば書体数をひととおり揃えてもそれほどの手間ではありませんし，書体の質も活字となんら変わるものではありません。対して，日本語ではまだ書体の種類も少なく，書体の質もいまひとつといったところです。

欧米では，たとえば，角の魚屋さんが作るチラシも新聞もほとんど同じクオリティが期待できるのです。魚屋さんとDTPを結びつけるもの，そこにはタイプライターとプレゼンテーションの文化が根底にあります。ときどき疑問に思うことがあります。個人でPageMakerなどを使っている日本のユーザーはいったいなにに使っているのでしょうか。

ハードウェアからの解放

フレームバッファを使わない「PC-9801での最高品質のグラフィック」を再現する場合，X68000では本来のグラフィック画面を使用するまでもありません。

すでにX68000のグラフィックでは解像度を補うアンチエリアシングの作法も浸透し，一部のソフトではRGBを24ビットカラーで内部処理しています。パソコンでは最高の高解像度65536色表示さえ，窮屈な制限にすぎないのです。ユーザーはそれを超え始めました。

しかしグラフィックなどよりもっともっとハードウェアの制限を受けているものがあります。それはテキスト表示です。もともとビットマップ表示というきわめて柔軟な構成を持って生まれたにもかかわらず，せいぜいがSX-WINDOWのエディタ.Xでさまざまなフォントに対応している（マルチフォントとはいえないが）程度で，テキスト出力環境はほとんどビットマップの恩恵を受けたことはありませんでした。

現在，X68000で使用可能な日本語フォントはROM内の16ドット，24ドットフォント，Z'sSTAFFなどで使われているアウトラインフォントです。

ただし，アウトラインフォントは一部のツールでサポートしているのみです。いくら高解像度のプリンタがあってもそれを支える土台がなければ「クッキリ鮮明な正方形の集合体」といつまでもつきあわなければなりません。

幸い，Z'sSTAFFのアウトラインフォントは書体倶楽部として拡充されつつあります。現在，Z'sSTAFFほか，CANVAS PRO-68K，TeX，さらにNEW PrintShop PRO-68Kver.2.0で使用可能，そう，シャープから発売されているソフトウェアでサポートされるということですから，X68000用アウトラインフォントの標準となることは間違いないでしょう。

これをあらゆるアプリケーションに展開していくにはもっと自在に，システムの一環として使用していくことができなければなりません（SX-WINDOWに“Zeit Type Manager”というのが出てくればいうことはないのですが）。

ピクセルから解放される日は遠いのでしょうか。

新製品紹介

IO-735X-B

今回新しく加わったIO-735X-Bは，これまでも発売されていたIO-735Xのブラックバージョンです。中身は特に変わっていません。Xシリーズ関係では唯一本格的なカラー出力に耐える機材として注目している人も多いことでしょう。

この製品はシアン，マゼンタ，イエロー，ブラックの4色カラーインクを扱うことのできる24ドット漢字カラーインクジェットプリンタです。

インクジェット方式とはノズルからインクを吹き付けて印刷するタイプのプリンタです。その特徴としては，印字が静か，比較的高速，印字品質が高いといったところでしょうか。最近BJ-10vで話題になっているキヤノンの「バブルジェット」というのもインクジェット方式のひとつと考えてかまいません。

インクジェットのランニングコストはドットプリンタと熱転写プリンタの中間といわれますが，少なくともあまり安くはつきません。このプリンタになんでもさせようとするとかなり不経済なことになります。カラーグラフィック出力専用と割り切って考えたほうがよいでしょう。

肝心のグラフィック出力ですが，このプリンタは4色のインクを使ってドットごとに標準的な8色の表示が可能です。アナログRGBで作成された多階調のグラフィックを出力する際はドットごとに色をコントロールして，平均で見て目的の色に近いものにしていかなければなりません。画面の解像度よりもプリンタの解像度が圧倒的に高いのでかなり自然に仕上がります。しかし，標準では打ち出しツールなどは付属していませんので，なんらかのユーティリティ（BANANA PRINT，Z'sSTAFFなど）を使うかプログラムを自作することになります。

シャープのIO-700/720/730シリーズはカラープリンタの標準的な存在であり，さまざまなソフトで対応がなされています。

印字品質についてはちょいと前のほうを見ていただければおわかりでしょう。

*　*　*

なお，CZ系列のプリンタではないのでプリンタドライバにPRNDRV.SYSは使えません（今回のカラーハードコピープログラムやテキストファイルの出力には支障ない）。コピーキーでハードコピーを取ったりする際には，プリンタをPモード（エプソンESC/P互換モード）にして，PRNDRV1.SYSを使うようにします。

JX-220X

200dpi（ドット/インチ）の解像度を持ったカラースキャナです。外見はひとまわりコンパクトになりましたが，性能的には従来のCZ-8NS1とほぼ同じものです。インタフェイスもソフトウェアもまったく同じで，ただし，従来は別売りだったパラレルインタフェイスが標準でついてくるのでお買い得になったといえるでしょう。元々，RS-232Cでも接続できたのですが，扱う情報量が膨大なため，パラレルインタフェイスは必須といってもよいものだったのです。

このクラスのスキャナはもはやRS-232Cで扱いきれるものではありません。パラレルインタフェイスボードでスロットを占有されるのもイマイチなので，SCSI対応の製品が期待されるところです。

元々，画質的にはほとんど無敵のスキャナだったのですが，さらに磨きがかかった印象があります。CZ-8NS1での唯一の欠点であった画面上部に色ムラができる症状もなくなっているようです。

面白いのは，JX-220XとIO-735Xを専用ケーブルで直結してカラーコピーのように使用できるということです。

50〜200%のズーム機能，2値化，モノクロ，ツールでの輝度補正や縦横独立変倍機能など機能も充実しています。グラフィックの入力装置としては「定番」といっていいでしょう。

IO-735X-B　248,000円
JX-220X　168,000円
シャープ　☎06(621)1221，03(3260)1161

ハードコピーの基本　　特集　印刷の世界へ

基礎からのカラー印刷

Hamazaki Masaya 浜崎 正哉

IO-730/735XとCZ-8PC1/2/3/4/5に対応したカラーハードコピーのプログラム例です。それぞれのプリンタで出力部分は違っても，基本的な部分は同じ。8色への変換にはSXIMAGEでもお馴染みの桒野式変換を使用します。

グラフィックのハードコピーを取るために必要な基礎知識をもう一度おさらいしておきましょう。ここではシャープ系の24/48ピンのカラー漢字熱転写プリンタとインクジェットプリンタ「IO-735X」を取り上げることにします。

前半はビットイメージの基本的な話を中心に，カラーハードコピーをするために多色画像をどうやって8色に変換し，印字させるかを説明していきます。最後にIO-735Xのカラーハードコピープログラムの解説をします。

プリンタの基礎用語

最初に，プリンタを使うために必要と思われる用語を解説していきます。

・印字ヘッド

プリンタが文字を印字する部分です。印字ヘッドにはピンがずらっと並んでいて，それが文字を印字していきます。ピンは，ある個数分，縦に並んでいてそのピンが右から左へ移動することによって印字がされることになります(図1)。よく，24ピンとか48ピンといわれるのは，印字ヘッドにあるピン数を指していてピン数が多ければ多いほどきれいな印字がされます。

・制御コード

プリンタにどういった動作をさせてやるか，を指定するためのコードです。制御コードをプリンタに送ることによって，プリンタが持つさまざまな機能を使うことができます。

・ビットイメージ

プリンタは文字コードを与えられるとプリンタ内のROMから文字パターンデータを取り出して印字していきます。たいていのプリンタではROMパターンだけでなくユーザーが指定したパターンを印字することができます。これがビットイメージ印字というものです。

パターンデータは印字ヘッドのところで説明したように縦の構成をとっています。印字に必要なデータは8ビットを単位としていてピン数によって一度に送るデータの数が決まります。24ピンなら3バイト，48ピンなら6バイトという具合です。

・改行幅について

改行幅というのは，文字どおり印字したあとにどれだけ行間をあけるか，ということです。改行幅を変えることによって，ハードコピーの場合には改行幅を詰めたり，ワープロなどで行間の調整ができるのです。で，改行幅は印字した下側から次に印字する行の先頭までの間のことではなく，印字した上側から次に印字する行の先頭までを指します（図2）。

・Y,C,Mとは？

これは，カラー印刷の3原色で黄色，シアン，マゼンタのことです。これらの色を微妙に混ぜ合わせてカラー印刷が表現されているわけです。

・データの転送

X68000でプリンタに直接データを渡したい場合には，

fp＝fopen("lpt","w")

でプリンタをファイルとしてオープンし，fputc( ), fwrite( )命令を使ってデータを転送します。よく使うコントロールコードなどは，あらかじめchar型の1次元配列に格納しておくと便利でしょう。

では，ビットイメージだ

今度は，具体的にどのようなデータをプリンタに送ってやるとビットイメージ印字ができるのかを説明していきます。それぞれのプリンタによってデータフォーマット

それなりに味のある3色分解

4色分解するとこうなる

が違うので，ここら辺はマニュアルを参考に話をしていきます。まずはビットイメージを行うためのコマンドを表１にまとめておきました。

ここでは代表的なプリンタ，そして一部のモードだけしか取り上げませんので，以下に記述されていないものについてはマニュアルをよく読んでください。プリンタごとだけでなく，使用するモードによってもデータの構造が違ったり，ビットの位置が違ったりしていますので注意しましょう。ではそれぞれどのようなデータ構成をしているのか説明していきます。

・「CZ-8PC3」24ドットビットイメージ

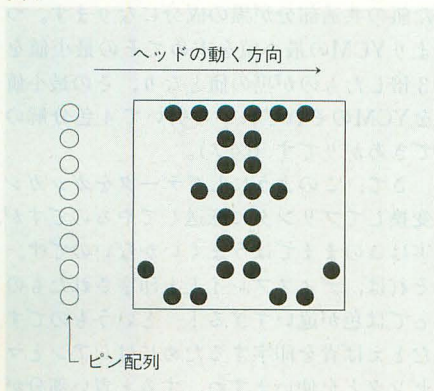
まず，図３を見てください。これはマニュアルの147ページにある説明を抜粋したものです。まずはビットイメージ印字を行うためのコマンド，次にデータの個数，そして印字するビットイメージデータがきます。データの個数というのはプリンタに送ったビットイメージデータの総数/3で求めることができ，16進２桁で表されます。つまり，横１ライン分のドット数と同じということでビットイメージデータの総数ではないことに注意しましょう。

データの並び順は，初めのほうで説明したように縦方向に３バイト（24ドット分）でビットの並びは図３にあるとおり，上が第７ビットに対応しています。

ハードコピーの手順は，変換した縦方向24ドット分（３バイト）のデータを横ドット分プリンタに送り，続けて“CR（$0D_H$）”，“LF（$0A_H$）”命令を転送していきます。それを縦のドット数/24だけ繰り返すようにします。

・「CZ-8PC4」48ドットビットイメージ

ほとんどCZ-8PC3の24ドットビットイメージと同じです。データ構成は図４のとおりで図２との違いはコードと縦方向に６バイト（48ビット）必要なことです。

・「IO-735X」カラービットイメージ

今回，IO-735Xでビットイメージを行うために制御コードグループGのカラーイメージ転送命令を使用することにしました。ちなみにIO-735Xには制御コードグループが３つ存在していて，それぞれ，

1) 制御コードグループG

カラーグラフィックプリントのアプリケーションを主目的とした制御コードグループ。IO-720，IO-735のGグループの制御コードをすべて使用できる

2) 制御コードグループN

漢字プリンタとしてのアプリケーションを主目的とした制御コードグループ

3) 制御コードグループP

主目的はグループNと同じ。AXパソコンの標準コマンドであるESC/Pモードに準拠していて，文章作成アプリケーションを対象にしている

となっています。

ハードコピーの手順は，４ライン分のデータを送ったあとに“ESC A（1/45インチ改行）”命令の転送を繰り返します。どういったデータ構造になっているかというと，ビットイメージを行うコマンド（２バイト），印字色とドットラインナンバー（１バ

表1

	CZ-8PC 3	CZ-8PC 4	IO-735X（Gモード）
ビットイメージ　8ドット	ESC% 2	ESC% 2	
ビットイメージ　16ドット	ESC I	ESC I	
ビットイメージ　24ドット	ESC J	ESC J	
ビットイメージ　48ドット		ESC M	
8ビットドット列リピート	ESC V	ESC V	
16ビットドット列リピート	ESC W	ESC W	
カラーモード設定	ESC EM	ESC EM	
n/120インチ改行	ESC % 9	ESC % 9	ESC T
カラーイメージデータ転送			ESC I
イメージデータ圧縮転送			ESC J
ドット列イメージデータ転送			ESC F
イメージデータストア			ESC S
イメージデータロード			ESC R

図2

図3　24ドットビットイメージ

コード：$1B_H$, $4A_H$, n_1, n_2, d_1, d_2, ………, d_k
$k=(n_1\times256\times n_2)\times3$

MSB b_7 〜 LSB b_0	d_1	d_4	…………	dk −2	1〜8
MSB b_7 〜 LSB b_0	d_2	d_5	…………	dk −1	9〜16
MSB b_7 〜 LSB b_0	d_3	d_6	…………	dk	17〜24

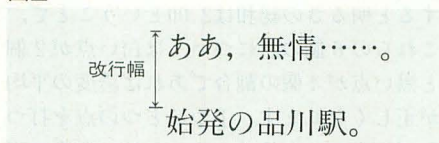
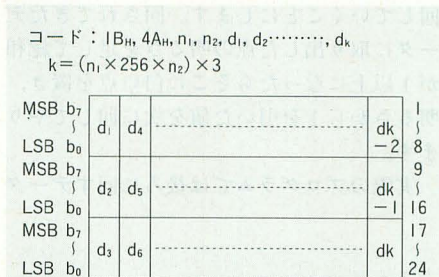



図4　48ドットビットイメージ

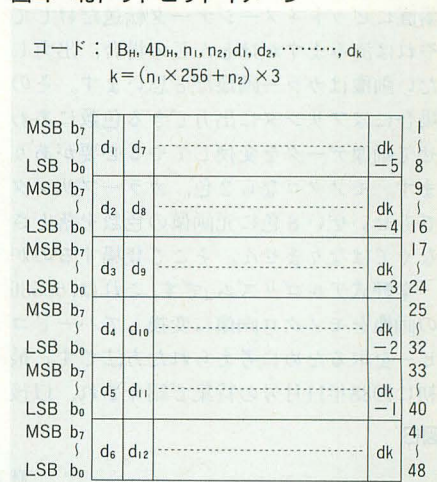
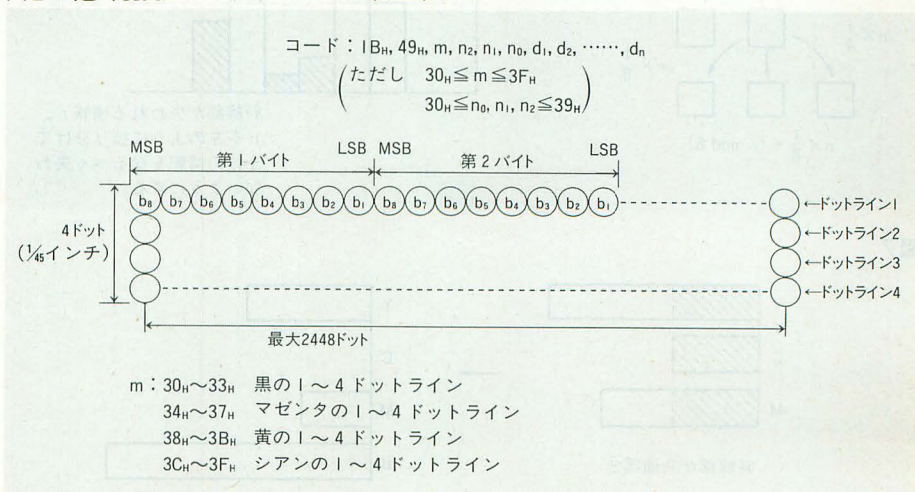



図5　IO-735Xカラーイメージデータ転送命令

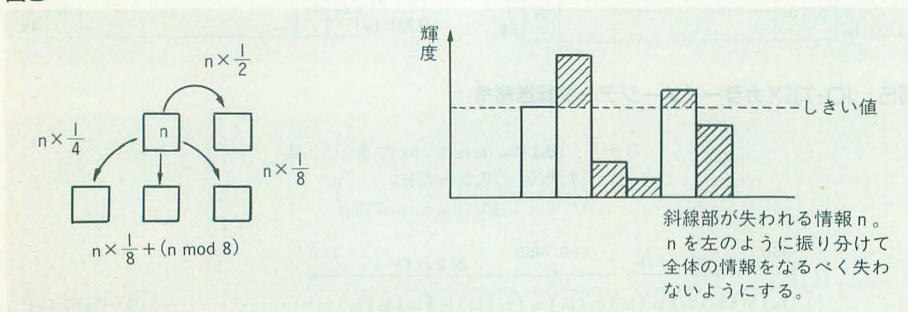
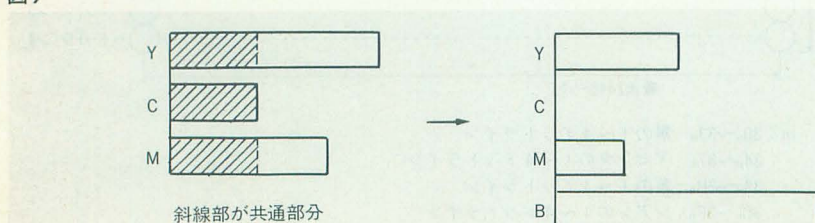
図1

イト），ドットイメージデータのバイト数（3バイト），ドットイメージデータで構成されています(図5)。ここでドットイメージデータは横を基準に並んでいることに注目してください。

で，そのドットイメージデータをどのラインに対応させるかを指定するのが「印字色とドットラインナンバー」です。プログラムでは横1ライン分のデータを変換し，それらのデータがどのラインに対応しているかラインナンバーを指定してプリンタに転送してやればいいことになります。データのバイト数は10進文字列で表します。238バイトのデータがあるときには，

32_H，33_H，38_H

の3バイトを転送してください。

桒野式アルゴリズム

以上の説明でビットイメージをする場合に，プリンタへどのようなデータを転送したらいいかわかってもらえたでしょう。ところで，モノクロ画像（2値化画像）なら素直にビットイメージデータ転送だけしてやれば済みますがほとんどの場合，出力したい画像はカラー画像だと思います。その場合にはプリンタに出力できる色数にあわせて画像データを変換してやる必要があります。モノクロなら2色，カラープリンタでもせいぜい8色に元画像の色数を落とさなくてはなりません。そこで登場するのが「桒野式アルゴリズム」です。これは，65536の画像をモノクロ画像に変換してハードコピーを取るために考えられた方法です。最初に1988年11月号の特集で紹介され，以後SX-IMAGEの16色変換でも使用されています。

アルゴリズムを簡単に説明すると，画面上の点がある一定以上の値を持つならば点を打ち，持たない場合には点を打たないようにしていきます（ある一定の値をしきい値と呼びます)。しかし，そのままでは画像から失われる情報が多すぎてきれいな変換はできません。そこで，しきい値までに足りなかった，もしくは余った情報を周辺のドットに振り分けることによって失われる情報を極力減らすようにしたのです（図6）。

要するに色ごとの明るさを数値化して，画面上の明るさの総和が正しくなるようにした，ということです。簡単な例を使ってもう少し説明していきます。まず，画面上の白いドットの明るさを1，黒いドットを0と考えます。するとすべてのドットは0から1までの値を取ることになりますね。たとえば画面上に6個の点があり，そのうち黒いドットが2個，明るさが0.30，0.60，0.10，1.00の点が1個ずつあったとします。すると明るさの総和は2.00ということで，これらの6個の点については白い点が2個と黒い点が4個の割合であれば濃度の平均が正しくなります。1つひとつの点を打つたびにまわりを見ていくことは，非常に面倒なので小数点以下のデータを次々後ろに回していくことにします。回されてきたデータに取り出した点の明るさを足して総和が1以上になったらそこに白い点を置き，明るさから1を引いた値を次に回してやります。

実際のプログラムでは後ろに回すデータを横だけでなく左下，右下，真下にも送っています。具体的にどのような値を振り分けるかというと，右横に8分の4，左下に8分の2，真下と右下に8分の1ずつで真下にはさらに8で割った余りを足した分，となっています。また高速化のためになるべく実数を使わず明るさを100倍して計算しています。

原理は簡単だしプログラミングもそれほど難しくはないのですが，しきい値をどれだけの値にするか，周辺にどれだけの割合で情報をばらまくのかが重要になります。桒野氏は試行錯誤でこれらの値を決めたようですが自然画だろうがアニメ調だろうが，なかなかきれいな変換がされます。

カラーのお話

いよいよIO-735Xを使ってカラーハードコピーを取ることを考えます。プリンタで使用できる色はYCMをベースにした8色ですから，画面上のBRGデータを，黄色，シアン，マゼンタの3色に変換してやる必要があります。変換手順は簡単でそれぞれビット反転をするだけです。そうして変換したデータを色ごとにプリンタへ転送していけばいいのですが問題は黒の扱いです。

熱転写プリンタでは，黄色，シアン，マゼンタの3色を混ぜると黒になりますがインクジェットプリンタの場合には焦げ茶色みたいな感じになり，しかもインクが滲んでしまいます。3色分解だとインクの滲みで印刷したものがすべて夕焼けのような風景になってしまうのです。そして黒っぽい絵などは無残にもでろでろにインクがたれてしまうことがあります。IO-735Xの場合には黒インクが使えるので，カラーハードコピーをしたいときには4色分解する必要があります。

では具体的にどのようにして4色分解していくかというと，まずBRGの値をビット反転してYCMの値を得ます。そして得られた値の共通部分が黒の成分になります。つまりYCMの最小値を求めてその最小値を3倍したものが黒の値となり，その最小値をYCMのそれぞれから引いて4色分解のできあがりです（図7）。

さて，このようにしてデータをガシガシ変換してプリンタに転送してやるのですが，実はこのままではうまくいかないのです。それは，ディスプレイ上と印字されたものとでは色が違いすぎる！　というものです。たとえば青を印字するためにはシアンとマゼンタとを使いますね，すると青い部分が

図6

図7

斜線部が失われる情報n。nを左のように振り分けて全体の情報をなるべく失わないようにする。

紫色に出てしまうのです。インクの関係上しかたがないことですが「印刷された絵が元画像と違うのは納得できない」と，中野修一氏が燃えに燃えてかなりのところまで近づけようと努力した結果は……これはカラーページを見ていただければわかるでしょうがだいぶ満足のいくものに仕上がりました。

プログラムの使用法について

リスト1がIO-735Xで65536色のグラフィックをハードコピーするプログラムです。ちなみにこのリストはコンパイルして使用することをお勧めします。インタプリタ上で動かそうというのは狂気の沙汰，試しに実行してみると印刷が終わるまでの間，完全に心地よい睡眠が取れることを保証します。

このプログラムを実行すると，まず印字するPIC形式のグラフィックのファイルネームを聞いてきますので，拡張子を省略してファイルネームを入力してください。

次に変換モードを聞いてきます。変換モードは，

・mode=0，YCMの3色分解して印字します

・mode=1，YCMに黒を混ぜた4色分解して印字します

・mode=2，mode=1に色調整部分を加えて印字します

の3つを用意しました。mode=0は最初，削除しようかと思いましたがインクの滲みでなかなか味のある印字になる，という利点があります。ディスプレイに忠実とはいえませんが，これはこれでいいんじゃない，ということでそのまま残しました。ちなみにmode=0を使う場合には，黒っぽいグラフィックは使わないようにしてください。また，インクの消費量がいちばん多いモードです。そしてmode=2はディスプレイにかなり忠実に再現されますがいちばん時間のかかるモードです。

今度は印字サイズの入力です。このプログラムでは一応印字サイズとして標準サイズ（688×512），ダブルサイズ（1376×1024），ドット比を1：1で打ち出す(1024×1024)，ポスターサイズ（3228×2448）を用意しておきました。特筆すべきはポスターサイズです。縦と横を逆転してめいっぱい印字させます。これは非常に印字がきれいです。広ければ広いほどまんべんなく拡散していくので遠くから見たらカラーコピーみたい，とまではいきませんがとても8色で打ち出したとは思えないはずです。インクに余裕のある人は，一度試してみるといいでしょう。ちなみにドット比1：1のモードはおまけみたいなものです。

印字サイズは縦方向，横方向ともプリンタが打ち出せる範囲内で任意の大きさに設定可能です。と，胸を張っていおうとしましたが実は制限があって，縦方向は4で割り切れる数値，横方向は8で割り切れる数値でなくてはなりません。

プログラムに手を加えれば何枚かに分けて印字してさらに大きなサイズを（時間とインクを惜しまなければ）印字することもできるはずです。ちなみに印字時間（mode=2，使用時）は，だいたい標準で17分，ダブルで44分，ポスターサイズで122分かかります。これは印字させる絵によって多少の違いがありますので目安程度として覚えておいてください。

リスト2は「CZ-8PC4」を使ってカラーハードコピーをするプログラムです。使い方はリスト1とほぼ同じで，色調整モードがないかわりに24ピン，48ピンモードの切り替えがつきました。24ピンモードだけならCZ-8PC3でもこのプログラムは使用することができます。

解説しよう

では，プログラムを説明していきましょう。だいたい私がプログラムを組みましたが色調整の部分は中野修一氏が担当しました。まずはプログラムの流れを見ていきましょう。

1) プリンタモードの初期化
2) 画面上のドットを取り出して，YCMB変換&色調整
3) 振り分ける値を計算
4) 色変換したデータを8ビット単位にまとめてバッファに格納
5) 4ライン分の変換が終了したら4色分それぞれのデータをプリンタに転送

リスト1

```
  10 /*
  20 /*  IO-730対応ハードコピープログラム
  30 /*
  40  dim int  line_buf(3,2500)
  50  dim int  ycmb_dot(3,2500)
  60  dim int  tonebuf(3,1,2500)
  70  dim char ycmb(3,3,2500)
  80  dim char gm(1) ={&H1B,'G'}
  90  dim char in(1) ={&H1B,'@'}
 100  dim char pp(1) ={&H1B,'4'}
 110  dim char kai(3) ={&H1B,'T','2','1'}
 120  dim int tone2(3),carry(3)
 130  char byte
 140  int H,S,V,C=-1
 150  int sizex=687
 160  int sizey=511
 170  int si,fp,mode
 180  str a,tl
 190     screen 1,3,1,1
 200     input "file_name:",a
 210     print "mode: 3 色 = 0 , 4 色 = 1 , 色 変 換 = 2 "
 220     input "mode:",mode
 230     si=15
 240     while si>3
 250       input "size(0:normal,1:double)?",si
 260     endwhile
 270       if si=0 then sizex=687:sizey=511
 280       if si=1 then sizex=1375:sizey=1023
 290       if si=2 then sizex=1023:sizey=1023
 300       if si=3 then sizex=2447:sizey=3227
 310     pic_load(a+".pic",0,0) /*または,apic_load(～),img_load(a+".gl3")
 320     tl=time$
 330     fp=fopen("lpt","w")
 340        fwrite(in,2,fp)
 350        fwrite(gm,2,fp)
 360        fwrite(kai,4,fp)
 370        fwrite(pp,2,fp)
 380          img_cnv()
 390     fclose(fp)
 400     for i=0 to 5 :lprint:next
 410      lprint tl
 420      lprint time$
 430      for i=0 to 5 :lprint:next
 440     end
 450 /*
 460 func img_cnv()
 470 /*  dim int carry(3)
 480   dim int tone(3)
 490   dim int col(3)
 500   dim int shikii(3)={3460,3420,3400,10400} /*Y,C,M,Bのしきい値
 510   dim int kido(3)={100,100,100,100}
 520   int xpos,ypos,dat,r,g,b,lc,lb,z
 530   int pcolor,x2,y2,dat2
 540 /*
 550   for z=0 to 3
 560    for ypos=0 to 1
 570     for xpos=0 to sizex
 580       tonebuf(z,ypos,xpos)=0
```

6)　縦の分が終わるまで2)から繰り返す

以上が基本的な流れです。次にサブルーチンごとに詳しく見ていきます。

1)　メインルーチン

ここでは，ハードコピーを取るPIC形式のグラフィックの読み込み，印字モードの入力，印字サイズの入力とプリンタの初期化を行っています。初期化する内容は以下のとおりです。

・プリンタリセット（プリンタを初期状態にする）
・制御コードグループをGに設定
・改行量を21/120インチに設定
・用紙フォーマットを電源投入時の状態に戻す

最後に5行分の改行を行い，印刷開始時間と終了時間を印字してさらに5行分の改行を行って終了します。

2)　img_cnv( )

データ変換のメインルーチンです。ここで使用している配列は全部で7本あります。それぞれどのように使われているかというと，

line_buf( )……YCMBのドットを打つかどうか格納
ycmb_dot( )……BRGからYCMBへ変換した1ライン分の変換バッファ
tonebuf( )……YCMBにおける左下，右下，真下に振り分ける値を格納
tone( )……YCMBの輝度を格納
shikii( )……YCMBのしきい値
carry( )……横の送り分を格納
tone2( )……輝度の一時格納用

となっています。ここでの流れは，

・送り先のtonebuf( )をクリアする
・ドットのカラーコードを取り出す
・取り出したデータをYCMに変換し，それぞれのデータをycmb_dot( )に格納
・黒成分の取り出しと色調整をする（black_get( )）
・それぞれの輝度を100倍する
・輝度に上のラインからの送り分と横からの送り分を足してtone( )に格納
・輝度の合計（tone( )）がしきい値を超えたらtone( )からshikii( )の値を引いてline_buf( )に1をセットする。超えなかったらline_buf( )に0をセット
・周辺に送るデータ量の計算
・1ライン分の変換が終わったらデータをプリンタに転送できるように横8ビット単位にまとめる（data_tran( )）
・4ライン分まとまったらプリンタに転送（img_print( )）

と，これだけの処理を縦のラインが終了す

```
 590      next
 600   next
 610   next
 620  for z=0 to 3
 630     carry(z)=0:tone(z)=0:col(z)=0
 640   next
 650 /*
 660   for ypos=0 to sizey/4
 670     for ycnt=0 to 3                      /*4ライン分のループ
 680     lc=ypos*4+ycnt and 1:lb=(ypos*4+ycnt+1) and 1
 690 /*
 700       for z=0 to 3
 710          for xpos=0 to sizex
 720             tonebuf(z,lb,xpos)=0
 730          next
 740         carry(z)=0
 750       next
 760 /*
 770       for xpos=0 to sizex
 780 /*
 790        x2=xpos*511/sizex
 800        y2=(ypos*4+ycnt)*511/sizey
 810         if si=3 then dat=point(511-y2,x2) else dat=point(x2,y2)
 820         if C<>dat then {                    /*同じ色なら色変換を省略
 830          C=dat
 840          dat=dat shr 1
 850          dat2=dat xor &HFFFF                         /*BRGをYCMに変換
 860             for z=0 to 2
 870                ycmb_dot(z,xpos)=(dat2 shr z*5) and &H1F
 880             next
 890          if mode > 0 then black_get(xpos,dat) /*黒の部分を抜き出す
 900          for z=0 to 3
 910            tone2(z)=ycmb_dot(z,xpos)*kido(z)
 920          next
 930         }
 940 /*
 950         for z=0 to 3
 960          tone(z)=tone2(z)+tonebuf(z,lc,xpos)+carry(z)
 970            if (tone(z)>=shikii(z)) then {        /*しきい値を超えたか?
 980                 tone(z)=tone(z)-shikii(z)
 990                 line_buf(z,xpos)=1
1000            } else {
1010              line_buf(z,xpos)=0
1020            }
1030           carry(z)=tone(z) / 8
1040           tonebuf(z,lb,xpos)=tonebuf(z,lb,xpos)+carry(z)+(tone(z) mod 8)
1050           if xpos > 0 then {
1060                       tonebuf(z,lb,xpos-1)=tonebuf(z,lb,xpos-1)+carry(z)*2
1070                 } else {
1080                       tonebuf(z,lb,xpos  )=tonebuf(z,lb,xpos  )+carry(z)*2
1090           }
1100           if (xpos<sizex) then {
1110                       tonebuf(z,lb,xpos+1)=tonebuf(z,lb,xpos+1)+carry(z)
1120                 } else {
1130                       tonebuf(z,lb,xpos  )=tonebuf(z,lb,xpos  )+carry(z)
1140           }
1150          carry(z)=carry(z)*4
1160         next
1170      next
1180       data_trn(ycnt,ypos):locate 0,10:print ypos*4+ycnt;"/";sizey
1190     next
1200      img_print()
1210    next
1220 endfunc
1230 /*
1240 /* 送られてきたデータをバッファに転送
1250 /*
1260 func data_trn(ypoint,yy)
1270 int z,x,dot,byte,dt
1280 int x2,y2
1290    dot=128:byte=0
1300      for z=0 to 3
1310        for x=0 to sizex
1320          if line_buf(z,x)<>0 then {
1330               byte=byte or dot
1340          }
1350         dot=dot shr 1
1360          if dot=0 then {
1370               ycmb(z,ypoint,x/8)=byte
1380               dot=128:byte=0
1390          }
1400        next
1410      next
1420 endfunc
1430 /*
1440 /* 変換されたデータをプリンタに出力
1450 /*
1460 func img_print()
1470     dim char img(5)={&H1B,&H49,0,&H30,&H36,&H34}  /*ドットイメージ
1480     dim char cr(1) ={&H1B,&H41}                  /*改行コード
1490     dim char data(2500)
1500     dim char im_col(3)={&H38,&H3C,&H34,&H30}
1510     int col,lin,dat,er
1520 /*
1530     beep
1540     img(3)=&H30+( ((sizex+1)/8) / 100)
1550     img(4)=&H30+( ((sizex+1)/8) mod 100) / 10
1560     img(5)=&H30+( (sizex/8+1) mod 10)
1570     for col=0 to 3
1580       for lin=0 to 3
1590        for dat=0 to sizex/8
```

るまで繰り返しています。

3) data_tran( )

1ライン分のデータ(line_buf( ))を8ビット単位にまとめてycmb( )に格納していきます。

4) img_print( )

ycmb( )に格納された4ライン分データをYCMBの順番でプリンタに転送。

5) black_get( )&colchange( )

ycmb_dotの配列に格納された1ライン分のYCMデータを元に黒の値の決定と色調整を行っています。色調整の詳しい説明は中野修一氏の記事を参照してください。

あとは色調整用に必要な，BRGをHSVに変換するサブルーチンだけです。このルーチンは1990年2月号51ページのリストを転用させてもらいました。

リスト2はCZ-8PC4に転用したものです。打ち出したハードコピーを見て気づいたことは，48ピンはドットが異常に細かい！　ということでした。そして，なぜか色のノリが悪いことにも気づきました。一応，熱転写用紙を使ってインクリボンも新しくしたんですけど……。24ピンモードではきちんと印字されているわけですからアルゴリズムはあっているはずだし，なぜだろう。リストの内容については，自力でがんばって解析してみましょう。

まだまだいけるぞ

さて，最後にこのプログラムの問題点を少々ばらします。まず，小さい絵の場合にゴミが目立つことがあります。どういうことかというと，少し暗いピンク色があったとします。そこにはわずかながら黒が混じっているわけです。するとプログラムはまばらに黒を置いていくので，なんか違和感を感じてしまうのです。回避方法として一番簡単なのが目立つ色のしきい値を上げるか，大きなサイズで打ち出す，もしくはそういうものだと納得することです（うっひゃあ)。

もうひとつは印字上部の部分をよ～く見ると，そこだけなぜかドットの並びが不自然なのがわかると思います。これは画面の一番上のラインには余分に送られてくるデータが存在しないためです。回避方法は……いい方法が浮かばないなあ。これは読者の皆さんも考えてみてください。

ほかにも色調整の部分がまだまだなんとかなりそうです。ま，65536色を8色に落とすわけですからこれぐらいが限度かな？と思いますがチャレンジ精神のある人はが

```
1600          data(dat)=ycmb(col,lin,dat)
1610        next
1620          img(2)=im_col(col)+lin
1630          fwrite(img,6,fp)
1640          fwrite(data,sizex/8+1,fp)          /*1ライン分の出力
1650      next
1660    next
1670      fwrite(cr,2,fp)                        /*4ライン出力してから改行コード
1680 endfunc
1690 /*
1700 /* RGBをHSVに変換
1710 /*
1720 func hsv_cnv(dat)
1730 int r,g,b,chk
1740   b=dat and &H1F
1750   r=(dat shr 5) and &H1F
1760   g=(dat shr 10) and &H1F
1770     V=max(r,g,b)
1780     chk=V-min(r,g,b)
1790       if chk<>0 then {
1800         S=(V-min(r,g,b))*32/V
1810         if r=V then H=(g-b)*32/(V-min(r,g,b))
1820         if g=V then H=(b-r)*32/(V-min(r,g,b))+64
1830         if b=V then H=(r-g)*32/(V-min(r,g,b))+128
1840           if H<0 then H=H+192
1850       } else {
1860         H=0:S=0
1870       }
1880 endfunc
1890 /*
1900 /* 最大値,最小値を求める
1910 /*
1920 func max(r,g,b)
1930 int x,y
1940    x=g+((r-g)+abs(r-g))/2
1950    y=b+((x-b)+abs(x-b))/2
1960     return(y)
1970 endfunc
1980 func min(r,g,b)
1990    return(-max(-r,-g,-b))
2000 endfunc
2010 /* 3色分解で出すだけならここから先はいらない
2020 /*
2030 /* 黒の部分を抜き出す
2040 /*
2050 func black_get(x,dat)
2060 int min,z
2070          min=999
2080          hsv_cnv(dat)
2090          for z=0 to 2                          /* 最小値の取り出し
2100            if ycmb_dot(z,x)<min then min=ycmb_dot(z,x)*9/10
2110          next
2120          if mode >1 then colchange(x,dat,min) /*色補正を行う
2130          for z=0 to 2
2140             if ycmb_dot(z,x)>=min then {    /*黒の値を引き算する
2150                ycmb_dot(z,x)=ycmb_dot(z,x)-min
2160              } else {
2170                ycmb_dot(z,x)=0
2180              }
2190          next
2200          ycmb_dot(3,x)=min*3
2210 endfunc
2220 /* 普通の4色分解でいいならここから先はいらない
2230 func colchange(x,dat,min)
2240 int z,chk,fff
2250       if (H>=96 and H<160) then {               /*青のチェック
2260         c=H-96
2270         ycmb_dot(2,x)=ycmb_dot(2,x)*c/63#+1
2280       }
2290 /*
2300        if (H>=32 and H<37) then {               /*緑の部分の色調整
2310         c=H-32
2320           ycmb_dot(1,x)=(ycmb_dot(1,x)*7*c/4#+2#)/15+0.2#
2330        }
2340        if (H>=37 and H<80) then {               /*緑本体
2350         c=H-37
2360           ycmb_dot(1,x)=(11+ycmb_dot(1,x))/2#*(6+(3*c/32#))/14#
2370        }
2380        if (H>=80 and H<96) then {               /*シアン接続部
2390         c=H-80
2400           ycmb_dot(1,x)=ycmb_dot(1,x)*(14+(10*c/15#))/28
2410           ycmb_dot(2,x)=ycmb_dot(2,x)*(8+(10*c/15#))/50
2420        }
2430                                                  /*青の部分の色調整
2440        if (H>91 and H<103) then {               /* シアン補正
2450         c=abs(H-96)
2460           ycmb_dot(1,x)=ycmb_dot(1,x)*((62#+c*2)/(12#+62#))
2470        }
2480        if (H>=96 and H<104) then {              /*シアン接続部
2490         c=H-96
2500           ycmb_dot(2,x)=(9+ycmb_dot(2,x)*2)/3#*(16+(3*c/7#))/22#
2510        }
2520        if (H>=104 and H<150) then {             /*青本体
2530         c=H-104
2540           ycmb_dot(2,x)=(20+ycmb_dot(2,x))/3#*(18+(3*c/45#))/20#
2550        }
2560        if (H>=150 and H<160) then {             /*マゼンタ接続部
2570         c=H-152
2580           ycmb_dot(2,x)=ycmb_dot(2,x)*((19+(9*c/11#))/21#)
2590        }
2600 /*
```

んばりましょう。

最後に余談ですが，ただいまOh!X編集部のマシンルームは完全にア○メイトしています(笑)。試し印字に趣味に走りまくったデータを使い，調子にのってポスターサイズで大量の印字をし，べたべた張っていたら自然とそうなってしまいました。おかげでほかの編集部から注目の的です（あんまり嬉しくないな)。そして床に転がる数メートルの無残なロール紙（失敗作品)。今度，ちゃんと掃除しなくちゃ。

参考文献

Oh!X 1990年11月号　特集「いまどきのプリンタ活用術」

CZ-8PC3，CZ-8PC4，IO-735X　取扱説明書

```
2610          if (H>=160 and H<192) then {                 /* ピンク?
2620              c=H-160
2630                ycmb_dot(0,x)=ycmb_dot(0,x)*c/32#
2640          }
2650          if (H>=0 and H<32) then {
2660              c=32-H
2670                ycmb_dot(0,x)=ycmb_dot(0,x)*(0.7#+c/32#*0.2#)
2680          }
2690 /*
2700 /* 輝度補正
2710          if (H>=96 and H<160) then {   /*青
2720                c=(H-96)*1.2#
2730                ycmb_dot(2,x)=ycmb_dot(2,x)*((10#+(S)*3#+c)/(62+10#+c))
2740                ycmb_dot(2,x)=ycmb_dot(2,x)*((31+5#)/(V+5#))
2750                ycmb_dot(0,x)=ycmb_dot(0,x)/2
2760                if ycmb_dot(2,x)>28 then ycmb_dot(2,x)=29
2770          }
2780          if (H>=32 and H<96) then {    /*緑
2790                ycmb_dot(1,x)=ycmb_dot(1,x)*((31+98#)/(S+98#))
2800                ycmb_dot(1,x)=ycmb_dot(1,x)*((31#)/(V))
2810                ycmb_dot(2,x)=ycmb_dot(2,x)/2
2820                if ycmb_dot(1,x)>28 then ycmb_dot(1,x)=28
2830          }
2840 endfunc
```

リスト2

```
  10 /*
  20 /*  CZ-8PC3,CZ-8PC4(24ドットモード)
  30 /*            対応ハードコピープログラム
  40 /*
  50  dim int  line_buf(2,7,2500)
  60  dim int  ycmb_dot(2,2500)
  70  dim int  tonebuf(2,1,2500)
  80  dim char ycmb(3,5,2500)
  90  dim char in(2) ={&H1B,&H63,&H31}
 100  dim char kai(3) ={&H1B,'%','9',16} /*改行幅の設定
 110  dim int tone2(2),carry(2)
 120  char byte
 130  int inji,pin,pb
 140  int C=-1
 150  int sizex,sizey,si,fp
 160  str a,tl
 170     screen 1,3,1,1
 180     input "file_name:",a
 190     si=15
 200     while si>3
 210       input "size(0-3)?",si
 220     endwhile
 230       if si=0 then sizex=687:sizey=511
 240       if si=1 then sizex=1375:sizey=1023
 250       if si=2 then sizex=1023:sizey=1023
 260       if si=3 then sizex=2447:sizey=3227
 270     inji=15
 280     while inji>1
 290        input "印字モード(0:24ピン,1:48ピン)",inji
 300     endwhile
 310       if inji=0 then pin=24:pb=2
 320       if inji=1 then pin=48:pb=5
 330    pic_load(a+".pic",0,0)
 340          /*または,apic_load(~),img_load(a+".gl3")
 350     fp=fopen("lpt","w")
 360       fwrite(in,3,fp)          /*プリンタ初期化
 370       fwrite(kai,4,fp)
 380        img_cnv()
 390     fclose(fp)
 400     for i=0 to 10:lprint:next
 410     end
 420 /*
 430 /* データ変換メインルーチン
 440 /*
 450 func img_cnv()
 460   dim int carry(2)
 470   dim int tone(2)
 480   dim int shikii(2)={3100,3000,3100} /*Y,C,Mのしきい値
 490   int xpos,ypos,dat,r,g,b,lc,lb,z
 500   int pcolor,x2,y2,dat2,ybyte=0
 510 /*
 520   for z=0 to 3
 530    for ypos=0 to 1
 540      for xpos=0 to sizex
 550        tonebuf(z,ypos,xpos)=0
 560      next
 570    next
 580     carry(z)=0:tone(z)=0
 590   next
 600 /*
 610  for ypos=0 to sizey/pin+1
 620   for ybyte=0 to pb
 630      for ycnt=0 to 7
 640      lc=(ypos*pin+ybyte*8+ycnt) and 1
 650      lb=(ypos*pin+ybyte*8+ycnt+1) and 1
 660 /*
 670         for z=0 to 2
 680            for xpos=0 to sizex
 690               tonebuf(z,lb,xpos)=0
 700             next
 710           carry(z)=0
 720         next
 730 /*
 740  for xpos=0 to sizex
 750    x2=xpos*511/sizex
 760    y2=(ypos*pin+ybyte*8+ycnt)*511/sizey
 770      if si=3 then {
 780           dat=point(511-y2,x2)
 790      } else {
 800           dat=point(x2,y2)
 810      }
 820       if y2>sizey then dat=65535
 830         if C<>dat then {          /*同じ色なら色変換を省略
 840           C=dat
 850           dat=dat shr 1
 860           dat2=dat xor &HFFFF            /*BRGをYCMに変換
 870              for z=0 to 2
 880                 ycmb_dot(z,xpos)=(dat2 shr z*5) and &H1F
 890              next
 900             for z=0 to 2
 910               tone2(z)=ycmb_dot(z,xpos)*100
 920             next
 930         }
 940 /*
 950   for z=0 to 2
 960    tone(z)=tone2(z)+tonebuf(z,lc,xpos)+carry(z)
 970                             /*しきい値を超えたか?
 980      if (tone(z)>=shikii(z)) then {
 990           tone(z)=tone(z)-shikii(z)
1000           line_buf(z,ycnt,xpos)=1
1010      } else {
1020           line_buf(z,ycnt,xpos)=0
1030      }
1040      carry(z)=tone(z) / 8
1050      tonebuf(z,lb,xpos)=tonebuf(z,lb,xpos)+carry(z)+(tone(z) mod 8)
1060     if xpos > 0 then {
1070       tonebuf(z,lb,xpos-1)=tonebuf(z,lb,xpos-1)+carry(z)*2
1080     } else {
1090       tonebuf(z,lb,xpos  )=tonebuf(z,lb,xpos  )+carry(z)*2
1100     }
1110     if (xpos<sizex) then {
1120       tonebuf(z,lb,xpos+1)=tonebuf(z,lb,xpos+1)+carry(z)
1130     } else {
1140       tonebuf(z,lb,xpos  )=tonebuf(z,lb,xpos  )+carry(z)
1150     }
1160          carry(z)=carry(z)*4
1170   next
1180   next
1190  next
1200        data_trn(ybyte)
1210  next
1220     img_print()
1230   next
1240 endfunc
1250 /*
1260 /* 送られてきたデータをバッファに転送
1270 /*
1280 func data_trn(ypoint)
1290 int z,x,y,dot,byte,dt
1300    dot=128:byte=0
1310      for z=0 to 2
1320       for x=0 to sizex
1330         for y=0 to 7
1340           if line_buf(z,y,x)<>0 then {
1350                byte=byte or dot
1360           }
1370            dot=dot shr 1
1380             if dot=0 then {
1390                ycmb(z,ypoint,x)=byte
1400                dot=128:byte=0
1410             }
1420          next
1430        next
1440      next
1450 endfunc
1460 /*
1470 /* 変換されたデータをプリンタに出力
1480 /*
1490 func img_print()
1500    dim char img48(1)={&H1B,'M'}    /*48ドットイメージ
1510    dim char img24(1)={&H1B,'J'}    /*24ドットイメージ
1520    dim char clor(1) ={&H1B,&H19}   /*カラーモード設定
1530    dim char img(1)
1540    dim char im_col(2)={0,2,1}
1550     int col,lin,dat
1560 /*
1570  fwrite(clor,2,fp)
1580  img(0)=(sizex+1) / 256
1590  img(1)=(sizex+1) mod 256
1600  for col=0 to 2
1610   if inji=0 then fwrite(img24,2,fp) else fwrite(img48,2,fp)
1620   fwrite(img,2,fp)
1630   for dat=0 to sizex
1640     for lin=0 to pb
1650      fputc(ycmb(im_col(col),lin,dat),fp)
1660     next
1670    next
1680     fputc(&HD,fp)
1690   next
1700   fputc(&HA,fp)
1710 endfunc
1720 /*
```

自然な色でのハードコピーを

HighFidelityへの挑戦

Nakano Shuichi 中野 修一

「画面の色とプリントアウトの色を近づけよう」とIO-735X用ハードコピープログラムに試行錯誤で色変換機能を追加しました。プログラムは93，94ページをご覧ください。ただし，最後まで調整しきれなかったようですが……。

美しいカラープリントを

編集室にやってきたIO-735Xを前にして，まずはBANANA PRINTを使ってみた。かなり綺麗なプリントアウトが取れる。予想以上だった。ディザ法を使っているなら「桑野式」にするだけでかなり解像度と階調が向上するはずだ，と目星をつけつつ，画面に32768色を表示したテストパターンを作って打ち出す（鬼のような奴）。うーむ，これはいけない。

確かに，ただの4色分解でも，理論上正しいだけあって，けっこう自然な変換を行う。しかし，原画（ディスプレイ）と並べるとどうしても違いの目立つものがある。

ただ絵が出ればいいというものでもあるまい。HighFidelity（高忠実性）は時代の必定。8色や16色の時代ではあるまいし，これではせっかくのアナログRGBが泣いてしまう。

幸いBANANA PRINTではかなり簡単に色変更ができるようになっている。青，赤，緑などの中間色（？）単位の指定もできるので，テスト印字の結果から傾向と対策を検討してもある程度は対応できる。それでも，ほかの色への影響は避けられない場合がある。全体的に原色を再現することは難しいようだ。

加色混合と減色混合

ディスプレイで見た色がプリンタでそのまま再現されるということを期待するのはきわめて自然なことではないだろうか。

今回の作業中に再現できない色に四苦八苦していると，まわりから「限界ですよ」と声をかけられる。私はなんの調整もなく，あらゆる画像に対して原色に近い色再生を実現することが可能だ，と思っている。ディスプレイのカラーハードコピーが同じ色で出力されるのは当然のことだろう。

しかし，現実はそうなっていない。

なぜだろう？　ディスプレイはRGB（赤，緑，青）の3色，プリンタはCMYB（シアン，マゼンタ，黄，黒）の4色を使って色を表現する。前者は光，後者はインクという違いがあるため，色を混ぜたときの結果は異なる。日常的な感覚では，赤と緑の光で黄を作るのもシアンとマゼンタを混ぜて青を作るのも多少違和感があると思う。これが加色混合と減色混合の違いだ。

絵の具（インクや透明水彩のようなものを想像してほしい）の3原色を混ぜると黒くなる。それに対し，光の3原色を混ぜ合わせると白くなる。これがもっとも基本的な違いといえる。

基本8色の対応は，

	R	G	B	C	M	Y
黒	0	0	0	1	1	1
青	0	0	1	1	1	0
赤	1	0	0	0	1	1
マゼンタ	1	0	1	0	1	0
緑	0	1	0	1	0	1
シアン	0	1	1	1	0	0
黄	1	1	0	0	0	1
白	1	1	1	0	0	0

のようになっている。ビット反転で簡単に対応できることがわかるだろう。印刷などではCMYに黒を加えたものが「フルカラー」として扱われる。

理論上，CMYの3色で黒が作れるのだが，実際には多色を混ぜると濁りが出るため黒を加えることになっている。プリンタのインクでも，すべてをあわせると暗い茶色になる。理論と現実のギャップは大きい。また，ディスプレイ上での100％の緑とインクでの100％の緑とでは明るさがまるで違ってくる。これも調整が必要となるだろう。

対応

ディスプレイ上での100％のRGBの輝度は，

B＝0.11
R＝0.59
G＝0.30

となるとされている。試みに，スキャナでYCMの3色を画像取り込みし，ディスプレイ上での輝度から明るさを推定して，インクの量を，ディスプレイ上での輝度の割合とプリンタでの輝度の割合で調整してみようとしたこともある。YCMのインク比率は一定のまま，輝度がディスプレイに忠実になるようにインクの全体量を変えるわけだ。かなり名案だと思ったのだが，YCMの輝度推定がいい加減なためか，ディスプレイ上の輝度が違うためか惨憺たる結果となった。

結局，落ち着いたのは色が違っている部分を選択的に最適化するという泥沼のような方法。

カラーページのグラデーションパターンを見てほしい。印刷の関係でちゃんと色が出ているかどうかはわからないが，かなり色が違うのがわかると思う。

とりあえず，ざっとテストを繰り返してみると，単なる4色分解では，

青が紫になる
緑が暗すぎる
ピンク色がオレンジがかる
Y，C，Mの純色が暗い
肌色がやや黄味がかる

という点が気になった。

これらを集中的に補正していこう。しかし，ある色だけを変えて，グラデーションの途中で色が不連続になるなどということが起きないように色成分はあくまで連続的に変更する必要がある。

あちらを立てればこちらが立たず

まず，ディスプレイ上の色をHSV（色相，飽和度，明度）形式に変換した。変換時の誤差が目立たないように，色相が分離された形式のほうが都合がよいからだ。今後の処理の大部分は，ある色相の範囲内に

限定して実行されることになる。

さて，色相というのは，

0　　赤
32　　黄
64　　緑
96　　シアン
128　　青
160　　マゼンタ

という具合に0～191のあいだに6色の原色が並んでいる（白と黒はS，Vの作用によるものだからここにはない)。

補正の概略をまとめると図1のようになる。実際にはもっと細かいところで変なことをやっているのであまり正確ではない。また，全体に純色部分の濃度を下げているので，飽和度が低かったり，輝度が低かったりするとかなりの誤差が出る。そこで輝度，飽和度での補正がさらに加わっている。主に手を加えたのは緑と青の部分だけで，あとの部分はそれほど大きな変更はしていない。

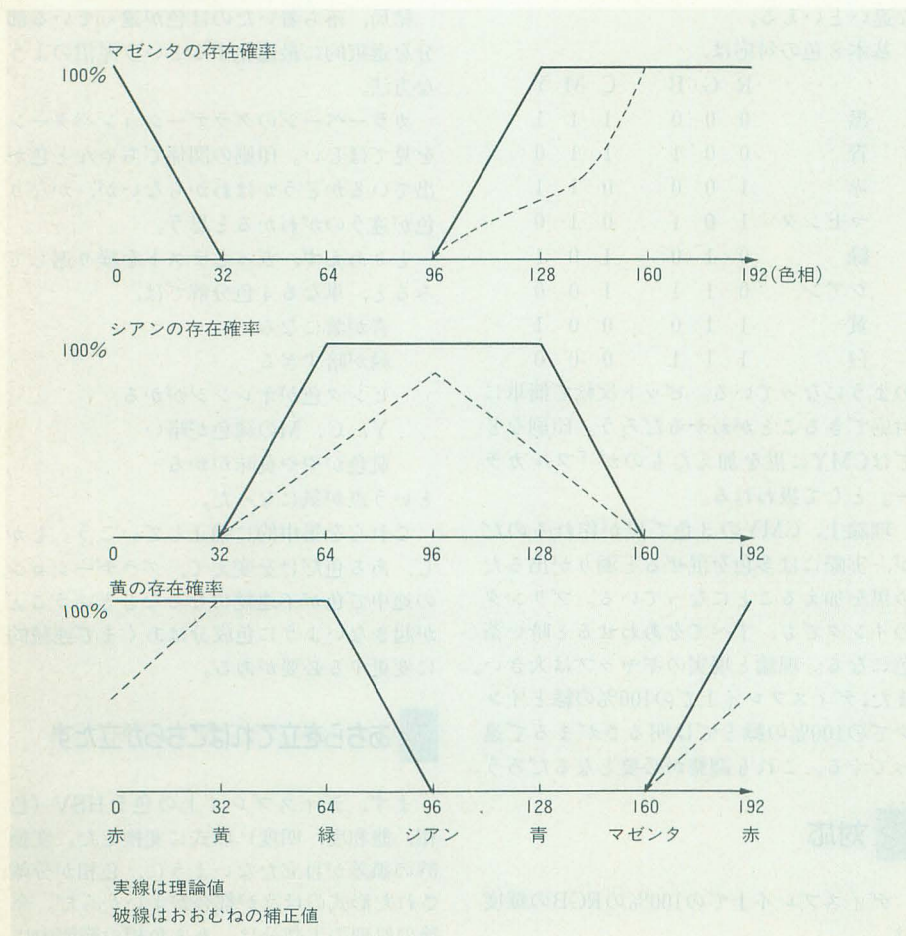
実際の処理はプログラムを見てほしい。ただし，対症療法の山なので「どうしてそこでこれを足し算しているのですか？」と聞かれても答えられない。定数値の意味はさらに希薄である。いくつかの画像で折り合いを取った結果がこのようになっただけで，ほかの画像ではまた再調整が必要かもしれない（かなりの画像で確認は繰り返した)。

全体の輝度や各色のバランスを変更したいときはしきい値をいじってみるといいだろう。しきい値を大きくすれば色は薄く，低くすれば色が濃くなる。そのほかの定数についてはバランスを見ながら調整してみてほしい。

図1　色補正の概略

熱転写への対応

熱転写プリンタでも同様のアプローチは可能なのだが，色のつき方がインクジェットとは異なること，インクリボンのA面とB面で色が異なるらしいこと，用紙へのインクの付着が均一でなさそうなこと，時間不足などの理由で今回は徹底的な最適化は見送ることにした。今回の色補正はインクジェットプリンタにあわせてあるので，そのまま使うとおかしなことになる可能性がある。特に熱転写の場合は青の補正はほとんどいらない（全体での補正ができれば）と思われる。

難物は48ドットビットイメージモード。48ピンの解像度は凄まじいのでうまくいけばきわめて高品質なプリントアウトが作成できそうだ。ただ，少し実験したところ，インクジェットより階調表現が劣っているようにも思われる。理論上は問題ないはずなのだが，うまく色がのってくれない。

インクジェットや24ピンの場合はドットの存在確率と濃度がリニアに変化していくのだが，48ピンになるとある程度薄い部分はほとんど白，ある程度濃くなると急に色ベタになってしまう。グラデーションを出すとほとんどデジタル8色になってしまうのだ。これを防ぐには中域だけを使用して確実に階調表現できる濃度を把握することから始めなければならない。これは用紙やリボンの向き，濃度調整などによってかなり変動することが考えられる。

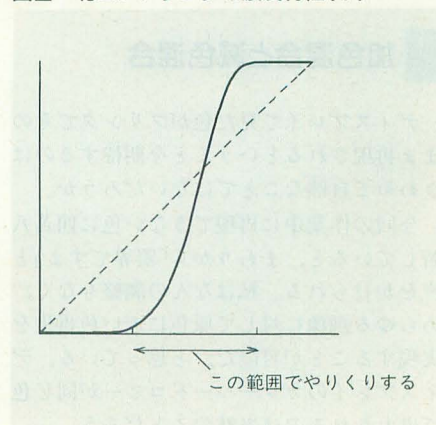
検討課題としておこう。

反省

色変換のおかげでずいぶん遅いプログラムになってしまった。同じ色が続くときは色変換をキャンセルするようにしたので少しはマシになったとはいえ，まだまだ遅い。ループ回数が画面ではなくプリンタ解像度に依存する（最大2448×3400）のと無頓着に実数演算を多用しているのが原因だ。

ちなみに，コンパイルにはできるだけGCCを使ってほしい。XCだと3倍くらい余計に時間がかかる。数値演算プロセッサも効果的かもしれない。XVIならさらによい（格段に速いぞ)。

しかし，これだけのことをやってもまだ満足できない色も残っている。やっぱり，32768×4個のテーブルを作るのがいちばんかもしれないなあ。

図2　48ピンプリンタの濃度特性(?)

ページ記述言語

PostScriptとはなにか

Tan Akihiko 丹 明彦

DTP関係ではなにかと耳にする「PostScript」。しかし，それがどんなものか知っている人はどれくらいいるでしょうか。ここでは言語としてのPostScriptの仕様の概略と基本思想を紹介しましょう。

PostScriptって?

現在，DTP環境を構築するうえでの大きな柱となっているPostScript（ポストスクリプト）。いろいろなところで話に出てくる単語だが，まだX68000とその周辺では見かけたことがないので大多数の読者の方にはピンとこないかもしれない。ちゃんと知っているのは，大学などでUNIXを使っている人くらいだろう。ここではページ記述言語PostScriptについて解説する。今回解説をすることが直接役に立つことはないだろうが，PostScriptの思想は知っておいて損はない。

＊

PostScriptをひと言でいえば，レーザープリンタで広く用いられている「ページ記述言語」となる。

レーザープリンタはページプリンタの一種で（以下の文では同じ意味で用いることにする），高品質かつ高速な印刷を行うことができるプリンタである。ドットインパクトプリンタや熱転写プリンタのような1行ずつ印字するプリンタと違い，用紙1ページ分（たとえばA4サイズの紙1枚分）を一度に印刷する。レーザープリンタの中身はコピー機にとてもよく似ている。行単位でなくページ単位で印刷を行うのでページプリンタと呼ばれる。解像度の高いものが多く，さらにドットプリンタや熱転写プリンタのようにプリンタヘッドの筋やむらが出ないため，印刷も美しい。

そのレーザープリンタで印刷するためには，レーザープリンタを制御するものが必要になる。

ドットプリンタの類だと，プリンタの機種ごとにコントロールコード（1バイトまたはそれ以上）が定められていて，通常の文字（アルファベットや数字などのキャラクタ）を送ったときにはそれを素直に印字し，コントロールコードを送ったときには文字印字以外の特殊な動作（たとえばグラフィック印字）をするようになっている。「コード××を送ったからプリンタは△△モードになって，そこに□□を送ると○○するので……」といったような制御のしかたをするわけである。

このコントロールコード（またはコード列）に字面上の意味はまったくなく，ハナモゲラといってもいい。さらにあろうことか，そのコードはプリンタによって異なるのである。したがって，同じ印刷をしたいときでもプリンタが違えばコントロールコードを変えてやる必要があるわけだ。数々のグラフィックツールやワープロなどの「環境設定」でプリンタの機種を指定させるのは，プリンタによってコントロールコードがまちまちだからである。困ったもんだ。

もちろん，多くのレーザープリンタは従来のプリンタと同じ命令を受け取るようにできている。シリアルプリンタをエミュレーションしているわけだ。しかし，これではページプリンタの真価は到底発揮できない。

PostScript搭載のレーザープリンタはその点異なる。PostScriptは一種のプログラミング言語だ。そのプログラムリスト(?)はエディタで書くことができる。これをレーザープリンタに送れば，そのプログラムリストそのものを印字する代わりにPostScript言語と解釈して実行（描画）するしかけになっている。言語仕様はどのPostScriptプリンタでも一緒だから，事実上共通のプリンタ制御言語として機能するというわけである。

むろん，コントロールコードを用いた場合に比べ，プリンタに送らなくてはならない文字の数は多くなる。しかし記述性の高さはそれを上回るメリットとなりうる。控えめに見ても，マシン語(ダンプリスト！)とPascalくらいの差はある。

文字だけでなく，直線や曲線などの図形を描くことができる。線の太さや濃さ，ラインスタイル(実線，破線，一点鎖線など)も指定できる。要するに，ドローイングツールで描ける程度のものは簡単に描くことができる。文字もさまざまなフォントの文字をさまざまな大きさで印刷することができるのだ。

Display PostScript

かのNeXTでは，紙だけでなく画面をもPostScript印刷の対象としている。それがディスプレイPostScriptである。画面の描画と紙への印刷を別に考えてプログラムする必要がないというのはありがたい。貧乏性の身としては，テスト印刷の代わりに使えそうなのもありがたい。紙を無駄に使うこともなく，便利。

PostScriptは別にプリンタ専用の制御言語ではない。あくまでページ記述言語である。ここにもPostScriptの思想の一端が見えるのだ。

PostScript言語の概要

PostScriptの言語としての大きな特徴に，スタック型言語だということが挙げられる。PostScript言語の動作は，引数をスタックに積んで，最後に演算子を与えるという形で行う。たとえば，

```
10 20 moveto
30 40 lineto
```

という記述は，「(10,20)へ移動し，そこから(30,40)まで線分を描画する」という意味になる。演算子（Cでいえば関数名）を引数の後ろに置いていることを除けば，ごくふつうの関数呼び出しと変わらない。

この「演算子後置」がPostScriptの名前の由来。“後ろに書く”からPost Script（ちなみに追伸という意味のp.s.はpost scriptからきているそうだ）。変数も持てるし制御構造も備えている。三角関数も手軽に使える（スタック型言語だから「60 cos」とやってcos(60°)を求める）。

C言語でいう関数定義に似たこともできる。ある手続きを定義しておいて，別の場

所から呼び出せるのだ。引数をつけて呼び出すことももちろん可能。気分は高級言語。ローカル変数が持て，再帰もできる。C曲線くらいならほんの数行のPostScriptのプログラムで書けてしまう。プリンタがこうも賢いと，使い方はいくらでも考えつきそうだ。

命令のうち，主なものを挙げてみよう。まずは描画から。

moveto

単純に移動する。

lineto

線分を描画する（本当は少し異なる）。

curveto

ベジェ曲線を描画する。

stroke

線を（実際に）描く。

fill

線で囲まれた内側を塗り潰す。黒色だけでなく灰色や白色で塗り潰すこともできる（このとき下の図形は覆い隠され，重ね描きにならない）。

※linetoやcurvetoは本当に線を描くわけではなく，strokeで線を描きfillで塗り潰す，その輪郭の形状を定義しているという雰囲気で捉えるとより正確。

setgray

strokeやfillなどの描画色を指定する。

ほかにももっと（細かい指定を挙げ出したらもっともっともっと）ある。なにしろマニュアルだけでぶ厚い本になってしまうくらいだから。

次は文字関係。

findfont

フォントの種類を指定する。英語フォントならローマン体を始めとして多種にわたるフォント(ギリシャ文字まである)，日本語なら明朝体とゴシック体がふつう。

scalefont

PostScriptプリンタの文字の大きさはポイントで与える。24ドットとか48ドットといった解像度を気にする必要はない。いわゆるベクトルフォントなので，どんな大きさの文字も出せる。1ページ大の文字を出しても，その輪郭は美しい（輪郭は多角形表現ではなく，曲線も使われている）。

setfont

カレントのフォントを指定する。

show

文字を書く。

……などとなっている。また，座標系も自在に設定できる。

translate

平行移動を行う。以後の描画は平行移動した新しい座標系で行う。

rotate

座標系を回転する。回転角度は自由。

scale

座標系を拡大縮小する。

そして，どしどし描き込んだところで，

showpage

ページを印刷する。

先ほども触れたが，ページプリンタは，現在のページに図形や文字を描いておいて，そのページをまとめて出力する。1ページ分の画像（印刷イメージ）を記憶しておいて，出力時にはそれをコピー機のように一気に印刷する。それを行うための命令がshowpage。いかにもページプリンタらしい命令といえよう。

＊

現在，パーソナルコンピュータ向けのレーザープリンタはいくつか出ているが，PostScript言語を搭載しているものはそれほど出ていない。代表的なのは例によってMacintoshのLaserWriter（Macっていつでも一歩先を進んでいる)。PostScriptに対応しているレーザープリンタには値が張るものが多い。独自のページ記述言語を搭載したプリンタもあるが，知名度や互換性などの点でいま一歩の感がある。

個人的には，計算機というものは，本体スペックに関しては個性を出してもいいと思うが，外部とのインタフェイス（たとえばケーブルやコネクタの規格には文句たらたらだ。同じ信号を送るケーブルなのに形状がまるっきりバラバラなものがこの世界にはいっぱいある）やデータのフォーマットに関してはある程度規格を固めたほうがいいと思っている（あくまで個人的には)。どうも現状は逆のような気がしてならない（本体に個性がないくせにインタフェイスだけが……)。プリンタのコード体系や制御方式がまちまちだと，困るのはユーザーなのだ。

現状ではDTPといえばMacintosh，という図式ができあがっている。高品質のプリントアウトを得るためには，出力の解像度を問わないPostScriptは必須条件といっていいだろう。Macintosh（MacintoshIIかな？）の高価なツール群のなかにはPostScriptを前提としたものがいくつかある。そこではスキャナから読み込んだデータをPostScriptデータに変換したりといったとんでもないことを平然と(……でもないか)やってのける。

PostScriptは思想的にたいへん優れたものを持っていると思う。単なるプリンタ制御言語に終わっていないのだ。そういうわけで，安価なPostScriptプリンタの出現を待望しているという次第（なんでも近々出るという話を聞いた)。その前に純正レーザープリンタをなんとかしてほしいという話もあるな。

PostScriptプリンタが高いわけ

日本語PostScript対応のページプリンタがある。日本ではまだ台数も機種も少ない。Macintoshを使っている企業やWindowsのPageMakerユーザーしか買わないからだ。なぜか。「高い」からだ。DOSマシンユーザーの場合「醜い倍角文字で満足している」ってのを加えてやろう。

日本語PostScriptプリンタは高い。6月現在，もっとも安いのが(なんと)日電のPC-PR3000PSで628,000円。7～8月にかけてブラザー（PostScript互換の独自言語）とかQMSが安いものを出すはずだが，それでも50万～60万円はするだろう。20万円台に突入したレーザープリンタの相場よりずっと高い。アメリカでは実売価格で$1,300（20万円以下！）というのもあるが，日本では，がんばっても50万円台だ。

なぜ高いのか？　まず，日本語PostScriptプリンタは内部にCPU，メモリ，ハードディスクを持っている。Apple社のNTX-J（798,000円）を例にとると，CPUに68020，メモリは2Mバイト，ハードディスクは40Mバイト。X68000より贅沢な環境だ。この分の値段は確実に上乗せ。

ハードディスクはフォントデータを入れるほか，データの展開用作業エリアとして使われる。PostScriptプリンタのメリットはここにもあって，フォントを増やしたいと思ったら，このハードディスクにフォントをダウンロードしてやればいいのだ。フォントのダウンロードはパソコン側で行う。フォントカードやカートリッジはいらないし，市販フォントも結構出ている。

続いて，アドビ社に払うライセンス料が高い。いまはいくらか安くなったという噂を聞くが，昔はプリンタ1台あたり10万円以上払っていたという話だ。何千文字の日本語フォントデータを起こすのにもかなりの金がかかっている。利用者が少ないのも，プリンタの値段が下がらない理由だろう。工業製品だからね。

DTPユーザーにとってPostScriptのメリットは計りしれない。PageMaker(MacintoshとWindows2.1)やQuark Express(Macintosh)といったDTPソフトが軒並みサポートしているからだ。MacintoshやWindows3.0のシステム自体がサポートしているのも大きい。さらに，1500～2500DPIという業務用のイメージセッタにはPostScript対応のものが多く，データを出力センターに持っていけば版下が作れてしまう。

ただ，PostScriptプリンタが完璧かというとそうじゃないのだ。PostScriptで記述できないグラフィックは描けないし，スタック言語であるから，1ページあたりのデータ量が多すぎると(つまり，文書が複雑すぎると)，プリンタのメモリが足りなくて印字できないこともあるからだ。これには驚いたね。　（荻窪圭）

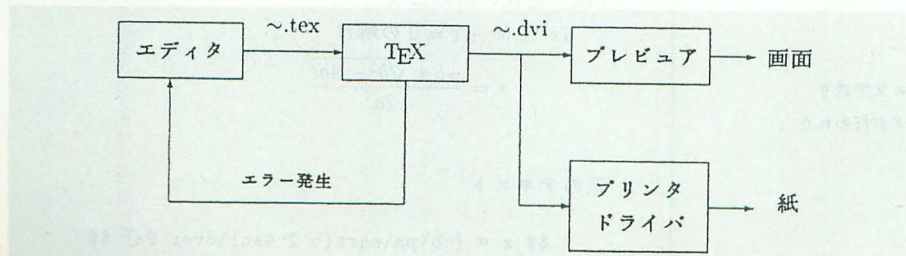
What you see is not ALL you get.

TeXからのアプローチ

Izumi Daisuke　泉 大介

理系大学生の友「TeX」。使いこなせば非常に強力な印刷システムとなります。以前からX68000で稼動していたソフトですが，安価な48ピンバルブジェットプリンタの登場で注目度急上昇。X68000で唯一のDTPソフトといえます。

少なからぬパーソナルコンピュータ上でDTPが進行しつつある状況を見ていると，標準添付のワープロが依然として主役の位置にある我らがX68000の環境はいささか貧弱であるといわざるをえません。AldusのPagemakerを筆頭とするDTPソフトの登場は，高品位な印刷物を自分で作成できる新しい環境を作り出しました。画面には印刷物のイメージが縮小して表示されており，それを見ながらタイトルをセットし，図版を貼り付けていく作業は，従来のワープロの世界とはまったく異なる環境を提供してくれます。

より細かい指定をしたい場合は画面表示を原寸大に，あるいは数倍にして作業することも可能になっています。タイトルや見出しをmm単位で移動させるといった，まさに組版レベルのことが簡単に実行できるようになっており，しかも，その効果を目で確かめながら実行できるというのがこれらのDTPソフトの大きな特長です。目で見たものがそのまま得られるという意味で，この方式はWYSIWYG (What You See Is What You Get：ウィジウィグ) と呼ばれています。

TeX (テフまたはテックと読みます) はこのWYSIWYG方式の対極に位置しています。作成する文書がどのようなイメージになるのかは，文書を印刷するまでわかりません。紙に印刷する代わりにディスプレイに印刷することもできますが，いずれにしても文書が完成するまでそのイメージを確認できない点は変わりません。これはTeXが，文字の大きさを何ポイントにするか，その行をセンタリングするのか右寄せにするのかなどといった情報を埋め込んだ文書ファイルを読み込み，一括して処理するバッチ方式を採用しているためです。

バッチ方式とTeX

普通のテキストファイル中に組版の情報を折り込むバッチ方式では，文書の作成から印刷までの手順は次のようになります(図1)。

1)　まず~.texというファイル名で原稿を作成します。
2)　それをTeXにかけると~.dviというファイルが作成されます。
3)　もし，途中でエラーが出れば原稿を手直しして再び2)からやり直します。
4)　めでたく~.dviファイルが作成されたら，これをプレビュアと呼ばれるソフトにかけて画面に表示するか，プリンタドライバと呼ばれるソフトにかけて紙に出力します。

通常はいきなり紙に出力することはせず，一度画面で全体のレイアウトを確かめてみます。たいていの場合はここで気にいらない点ができ，手直しすることになるからです。もちろん，誰かが作った~.texファイルを印刷する場合には，それを画面で確認する必要はありません。

画面上で実際の効果を確かめながら作業できるWYSIWYG方式に比べると，マジックハンドで遠隔操作しているかのような印象のあるバッチ方式ですが，この方式の最大の魅力は，手慣れたエディタで文書の作成ができる点にあります。もちろん，WYSIWYGでもエディタやワープロで作成した文書を流し込むことは可能ですが，TeXでは書式に関する指示もすべて手慣れたエディタ上で与えることができるのが大きな違いだといえるでしょう。たとえば書体をcmcsc (小文字サイズの大文字) に変更したいなら，

¥font¥testfont＝cmcsc10
¥testfont
What You See Is ALL You've Got.

のように指示します。これは10ポイントのcmcscフォントをtestfontという名前で使うことを宣言し，実際にtestfontを使っているところです。＼ (バックスラッシュ) が使えない場合は，代わりに¥で指定します。印字結果は図2のようになります。

この様子は，本誌でプリンタ特集のたびに掲載されたある種のプログラムを連想させるかもしれません。もともとプリンタはパイカやエリートなどの書体を備えており，エスケープシーケンスと呼ばれるコントロールコードを使って，書体を変えたり，上付きにしたり下付きにしたり，あるいは横倍角，縦倍角，4倍角にすることができます。エスケープシーケンスは特殊な文字列なので直接文書中に埋め込むことができません。そこでBASICのLPRINT命令を駆使した，プリンタに自由に文字を出力する実験プログラムが登場することになるわけです。

これはさすがに面倒なので，次のようなアプローチもあります。上付きにしたい場合には文書中に「¥super」の文字を埋め込んで指示するというルールを決め，プログラムでこの文字列を実際のエスケープシーケンスに変換してプリンタに送るのです。実際私自身，字詰めを決めて文書を整形するプログラムを作成したときに，プライベートにこの機能を付加して遊んだこともあ

図1　作業図の流れ

りました。

いささか乱暴ですが，TeXは後者のようなプログラムを大幅に機能拡張させたものと見なすことができます。出版物と同じレベルの印刷を行うために書体は非常に豊富に揃えられていますし，改行幅も自由に設定できるようになっています。さらには何cm×何cmの範囲に印刷しなさいという設定も行うことができます。もちろん，AVという文字列が現れた場合に間をつめたり（カーニング），ff, fiといった文字列をつないで表示する（合字），2行にまたがる単語は途中にハイフンを挿入して単語が生き別れになることがないようにする（ハイフネーション）などといった英文を扱う上での基本的な機能もちゃんとサポートされています（図3）。

さて，このcmcsc10という書体を章の名前を表示するのに使うことにしたとします。どんどん書き進んで文書が完成し，全体を眺めてみてやっぱりcmcscはよそう，などという状況になったときを想定してみましょう。図2では最初の行で使用するフォントを宣言し，空白行のあとの第4行目で実際にそのフォントを使用しています。このため，文書の先頭にある¥testfontの指定を変更するだけで，¥testfont書体を使って書かれているはずの章の名前はすべて自動的に変更されることになります。

ワープロではこうはいきません。タイトルを文書中から探しだし，片っ端からプルダウンメニューの書体変更で書体を変えていくことになります。WP.Xのポップアップメニューのように「AGAIN」のような機能が備わっていたとしても，これはなかなかに面倒な作業です。一発で該当する書体をすべて変更できるのは，バッチ方式ならではのメリットです。最近ではWYSIWYG方式のDTPソフトでも，章名の表示形式を「スタイル」として登録しておき，スタイルの変更だけですべての章名を変更することができるようになっています。バッチ方式のよいところを取り入れた例といえるでしょう。

数式自由自在

見たままのものが得られ，直感的に操作しやすいという魅力ある特長ゆえ，WYSIWYGは一般に広く普及したDTPの方式です。にもかかわらず，バッチ方式であるTeXが圧倒的に使用されている場所があります。大学の理系学部，そして理系の研究所などです。これは，TeXがWYSIWYG方式のDTPソフトにはない大きな特長を持っているからだと思います。

TeXのもつ特長，それは数式を扱う能力に長けていることです。図4をご覧ください。これと同じ出力をワープロで得ようとすると，半分改行や上付き文字，文字列のセンタリングなどの技法(?)を駆使しなければなりません。WP.Xでは，単にbの2乗を作るだけで，bと入力したあとで1/4角文字をプルダウンメニューから選択し2を入力するという手順が必要となります。多くのDTPソフトも，文書と図版や表などのレイアウト機能に重点を置いているため，このような出力を得ようとすると少なからぬ努力を強いられることになります。

TeXでbの2乗を作成するのは簡単です。数式モード（$で文字列をくくるだけ）で，

```
$b^2$
```

とすればOKです。a分のbなら，

```
$b ¥over a$
```

となります。aの平方根なら，

```
$¥sqrt a$
```

です。上付き文字を指定する必要も，半分改行を駆使する必要もありません。数式を別組みにしたいなら，$$で数式をくくります。そしてこれらの機能を組み合わせれば，見事，図4の出力が得られるわけです。

累乗や分数の指定は，再帰的に繰り返し使うことができます。図5に分数を分母に持つ分数の例を用意しました。この例を見て気付くことは，TeXが「1/b+1」の部分を縮小して分母としていることです。分母の文字を縮小せず，通常の表示に使う文字で出力することも可能なのですが，分数のバランスを考えてこのような表示方法が標準で採用されています。この分母部分にも分数を使うと，書体はさらに小さなものが使用されます。

数式を扱う分野，特に数学の分野では，一般には使われない（JIS第1水準，第2水準に含まれない）記号を使うことがよくあります。たとえば無限を表す〔ℵ〕は漢字ROMでは扱うことができません（もちろん外字として定義することはできますが）。このような数学記号の例を図6に挙げておきます。

数式を手軽に記述でき，必要な記号が揃っていて，しかも全体を整形して綺麗に出力できる。これは理工系の研究者にとって

図2 cmcscフォントでの出力

元のテキスト

```
\font\testfont=cmcsc10
What You See Is What You Get;but remember,
\testfont
What You See Is ALL You've Got.
```

TeXによる出力

What You See Is What You Get;but remember,
WHAT YOU SEE IS ALL YOU'VE GOT.

図3 カーニングと合字の例

元の文字列		TeXの出力	
AA	⇒	AA	ノーマルの文字送り
AV	⇒	AV	カーニングが行われた
ff	⇒	ff	合字の例
fi	⇒	fi	
fl	⇒	fl	

図4 簡単な数式の例

2次方程式の解の公式

$ax^2+bx+c=0$ の解は

$$x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

元のテキスト

```
$$ x = {-b\pm\sqrt{b^2-4ac}\over 2a} $$
```

は大きな魅力です。TeXが数式の扱いに長けているのは，クヌース教授が自分の数学の本を出版するための組版ソフトとしてTeXを作り出したためです。ほんのちょっとした数式でも，写植屋さんは見事なまでにメチャクチャにしてくれることがあります。それに初校で赤を入れて再校をとると，またまた違った形にレイアウトされ，それを直して念校までとっているのに結局は出張校正で直さなければならない……。欧米では数式の組版のできる技術者が圧倒的に不足しているということですが，なかなかどうして，日本でも大変です。TeXを使えばすべて自分の思いどおりにできるのですから，重用されるのも道理といえるかもしれません。

確かにTeXで原稿を作るのに要するタイプ量は少なくないのですが，これらの研究者の中には手で書くよりタイプしたほうが速いという人が少なくないのも大きな理由といえるでしょう。分数を表現するのに半分改行やセンタリングを考える必要もなく，「a ¥over b」でいい，というのは魅力的です。

充実のフォント群

TeXは数学記号を豊富に備えているだけではありません。ローマン，太字のローマン，小文字サイズの大文字，イタリック，タイプライタなど，さまざまな書体が用意されています。太字のローマンはワープロの強調文字のように単に横にずらし重ねて出力することによって太くするものではなく，最初から太字のローマンとしてデザインされています。これらのフォントの一部を図7に挙げておきます。

先の数学記号やこれらの書体が外字と決定的に異なるのは，種類だけでなくサイズも豊富にあるという点です。分数の例では，分母の分数に若干小さなフォントが使われていました。これらの書体の大きさの指定にはポイントという単位を使います。12ポイントはパイカ1文字の大きさで，パイカは1インチ（2.54cm）に6文字打つことができます。

日本語ワープロの出力で，タイトルを強調にするだけでは飽きたらず，4倍角のしかも斜体にするという例をよく見かけます。技法を駆使したというよりは，できることを（吟味なしに）すべて施したという感じです。本文は10ポイント，章名などのタイトルは17ポイントや14ポイントという出力を見てしまうと，ワープロの出力はいかにも貧相で，センスのかけらも感じられなくていけません。幸い日本語のアウトラインフォントが徐々に整備されつつあるので，近い将来には過去の笑い話となることでしょうが。

これらTeXで使用されるさまざまなフォントは，METAFONTと呼ばれるTeX用のフォント作成システムで作られたビットマップフォントです。クヌース教授はTeXシステムを10年の歳月をかけて完成したそうですが，そのうちの7年をMETAFONTシステムの作成と，そのフォント作りに費やしたということです。長い時間をかけて作成されたこれらのフォントはcmフォント（Computer Modernフォント）と命名されています。

METAFONTはフォントを記述したファイルを読み込み，指定されたフォントをさまざまなポイント数の，さまざまな解像度のビットマップフォントとして出力できるようになっています。なぜベクタフォントではなくビットマップフォントを使うのか疑問をもたれるかと思いますが，おそらく処理速度を考慮してのことではないでしょうか。

METAFONTシステムの存在によって，ユーザーは自分が必要とする記号を自分の手で作り出すことができるようになりました。数学記号が充実しているのもこのためですし，自社のロゴをMETAFONTを使って作成することも可能となりました。さんざん書かされたレポートのタイトルページに，大学のロゴが燦然と輝いていたのを思い出します。

強力なマクロ

TeXが理系の分野で支持されているもうひとつの理由は，TeXがきわめて柔軟なシステムであることだと思います。TeXの核は，組版を行うための非常に細かい命令の塊のようなもので，これだけを使って目的の文書を仕上げるのは困難です。TeXはこれらの細かい命令をいくつかまとめて，センタリングをする命令などの一般的な命令としています。ミクロな命令を組み合わせてよりマクロな命令を作り出すこの方法は，マクロ定義と呼ばれています。

TeXはこのマクロ定義をユーザーに開放しています。10ポイントのcmcscフォントをtestfontという名前で使うと宣言したのは，このマクロ定義機能を使っている例です。cmcscフォントを章名だけでなく，文章中の強調したい部分にも使いたい場合があるかもしれませんので，「¥chapter」などという名前のマクロを定義し，左寄せして14ポイントのcscscで表示する，などと指示したほうが現実的です。章名の形式を変更したくなった場合にも，これならほかのcmcscフォントで書かれている部分に影響を与えることなく変更が可能となります。

一般に使用されているTeXシステムは，TeXの核にplainマクロと呼ばれるファイルで提供されるマクロパッケージを導入したものです。マクロは上のような考え方で定義されており，ユーザーが使うTeXの命令の大半はこのマクロパッケージで提供されたマクロです。

マクロパッケージで提供されたマクロを使ってさらに自分専用のマクロを作ることで，自分に使いやすいシステムに仕上げていくことができるというのは実に魅力的な機能です。もちろんマクロを作るためにはマクロ定義のしかたを勉強しなければなりませんが，一種のプログラミング言語のようなこのマクロは理系の人々にとって与しやすいものだといえるでしょう。そして自分好みの設定を容易に行えるからこそ，彼らはTeXを好むのではないでしょうか。

マクロがユーザーに開放されていることは，別の面のメリットをもたらします。章名を書くときは「¥chapter」を使って，

¥chapter {章名}

のようにしなさい，というルールを決めることによって，誰が書いても同じ体裁のレポートとすることができるのです。章の下

図5　分数の中で分数を使う

$$\frac{1}{a+\frac{1}{b+1}}$$

元のテキスト

```
$$ 1 \over a + {1 \over b+1} $$
```

図6　数学記号の例

$\llbracket \quad \approx \quad \sqsubseteq \quad \oint \quad \ll \quad \succeq \quad \simeq$

図7　さまざまな書体の例

This font is roman.
This font is bold roman.
THIS FONT IS CAPS AND SMALL CAPS.
This font is text italic.
`This font is typewriter.`

のレベルのセクションを書くときには,

¥section {セクション名}

とする。さらにその下のレベルは,

¥subsection {サブセクション名}

とする……，としておけば，レポートの内容を追いかけていくのも容易になります。

このようなメリットを際だたせているのがLaTeXと呼ばれるシステムです。これはTeXのコアにlplainというマクロパッケージを導入したもので，文書の書きやすさと読みやすさを考慮したシステムになっています。たとえば分数は,

¥frac {分子} {分母}

の書式で表記します。これは「¥over」を使用したものより分子分母がはっきりしているので書きやすいスタイルです（とくに複雑な分数式ではありがたい)。体裁を整えるという面では，ドキュメントスタイルの設定という方法が用いられています。文書の先頭に,

¥documentstyle {スタイル}

と書くことによって，章立てやセクション分けなどが前述のようなマクロによって行われるようになっているのです。論文形式，レポート形式，本の形式，手紙の形式などのスタイルが標準で用意されています。これらのスタイルにさらに手を入れて，独自のスタイル，たとえばOh!Xスタイルなどを作り出すことも可能です。

さまざまなマクロを定義されたTeXは，まったく新しいTeXシステムを簡単に作り出します。LaTeXとともに有名なAmS-TeXは数学の論文用のTeXシステムで，数学で使用するさらに多くの記号（大半は見たこともない）が定義され，自由に使えるようになっています。

X68000とTeX

このように強力な整形システムで日本語が使えたら……，という要望は多く，現在では日本語TeX，JTeX，JLaTeXなどが使えるようになっています。

また，パーソナルコンピュータへの移植も盛んに行われており，我らがX68000でも通信などで入手すればJTeXが利用できます。JTeXさえ動いてしまえば，あとはマクロパッケージを導入するだけですから，JLaTeXでDTPを楽しむこともできます。現在ではTeX，LaTeX，JTeX，JLaTeXのいずれもが利用できるようになっています。X68000はワープロのラインアップすら少なく，さらにその上をいく市販品のDTPソフトとなるとまったく影も形も見えません。そんななかでTeXは実用になる唯一のDTPソフトといえるでしょう。

TeXの日本語化に伴う問題点として，日本語フォントのインプリメントという作業があります。英数字のほうはMETAFONTで作ったフォントが付属していますが，日本語のMETAFONTはないのです。ワークステーションなどでは，大日本印刷が破格値で提供してくれている秀英フォントが利用されています。ただ，いくら安いといっても（1セット98,000円）個人で購入するにはまだ高いというのが現状でしょう。かといって，ROMの24ドットフォントが拡大縮小されて出力されるのでは，いくらTeXが綺麗にレイアウトしてくれても興ざめです。

X68000用のTeXをインプリメントした方々はうまい方法を採用しました。Z'sSTAFFのベクタフォントを採用したのです。CANVASPRO-68Kでもお馴染みのZ'sSTAFFのベクタフォントは明朝，ゴシックの両方の書体を備えており，フォントデザインもすっきりしています。当然ベクタフォントなので，何ポイントに拡大しようとギザギザになることはありません。また，この部分はフォントドライバとして独立していますので，将来フォントが増えたときにも容易に対応できることでしょう。

ベクタフォントを持っていない人のために，ROM内フォントと，それを拡大・スムージングしたフォントを利用できるようになっているのも優しい配慮です。

レポートや論文などを，自宅のX68000で納得いくまで試行錯誤できるというのは，とかく混みがちなワークステーションの端末では味わえない楽しみです。好きなだけ専有できるパーソナルコンピュータのメリットを存分に生かして納得したテキストを作ったら，それを自宅のプリンタで出力するもよし，大学や会社のワークステーションの高解像度のレーザープリンタで出力するもよし，です。

X68000用のTeXを使うには，2Mバイト以上のメモリと，10Mバイト以上のハードディスクの空きが必要です。TeXがビットマップフォントを扱うため，圧縮されているとはいえフルセットのフォントファイルは大容量で，これがフロッピーディスクでの運用のネックになります。

拓かれた360dpiの世界

実用になる整形システムがあり，実用になる日本語フォントがあれば，次にほしくなるのは実用になる高解像度プリンタです。やはりここは，安くて綺麗なページプリンタの登場を望みたいものです。Xシリーズは伝統的に独自のプリンタコントロールコード体系を採用していますが，ページプリンタではXシリーズローカルから抜け出してほしいと思います（CZ系列の24ピンプリンタのエミュレーションすら載せるなとはいいませんが)。自社開発のPostScriptコンパチなら文句のないところです。解像度は最低でも300dpiはほしい。値段は20万円を切ってほしいと，ユーザーの勝手な要望はとどまるところを知りません。

ページプリンタだけが高解像度プリンタではありません。安価な熱転写プリンタといえど，CZ-8PC4/8PC5などの48ピンプリンタになると，その解像度は360dpiに及びます。現在市販されているレーザープリンタの多くが240dpiの解像度しかないことを考えると，その能力はあなどれません。

キヤノンから発売されているBJ-10vは360dpiの48ドットインクジェットプリンタながら，定価74,800円と非常に魅力的な価格設定がなされています。実売価格は6万円を下回っており，現在もっとも安価に購入できる高解像度プリンタだということができるでしょう。コントロールコードはESC/Pを採用しており，PRNDRV1.SYSを使用することで，X68000で使うことができます。

私はこれをTeXの出力に使っています。今回の図版はすべてX68000＋JLaTeX＋BJ-10vで作成したものを縮小して掲載しています。いかがですか，結構いい線いってると思いませんか。高解像度だとはいえシリアルプリンタであることには変わりありませんから，多少ズレが生じている場所はありますが，240dpiのレーザープリンタよりは優れた出力です。

ただ，問題がないわけではありません。TeX用のプリンタドライバはビットイメージ印字を使ってプリントアウトを行います。ところが，BJ-10vのビットイメージ印字のコード組成が一般のプリンタと異なっているのです。大半のプリンタは実際に転送するビット数をピン数で割った値をデータ量として渡します。BJ-10vの24ビットイメージモードもこれに従っています。ところが，48ビットイメージモードだけは転送するビット数を8で割った値を用いるのです（このコードはESC/Pではありません)。私の持っているTeX用のプリンタドライバはこのタイプに対応していないため，そのままではBJ-10vに出力できませんでし

た。結局プリンタドライバを逆アセンブルしてパッチを当てたのですが，一般のユーザーの方には少々辛いところです。BJ-10vが発売されてからだいぶたちますから，現在ではプリンタドライバがバージョンアップされている可能性はありますが……。

熱転写プリンタでビットイメージ印字を行うと，印字するドットがない場合でもインクリボンは先に送られてしまいます。インクジェットプリンタでは，消費されるインクは印字したドット分だけですから，ランニングコストも熱転写プリンタより抑えることができるでしょう。初期投資が安く，ランニングコストはそこそこで，印字品質は高い。BJ-10vによって，本格的な360dpi時代が始まりそうです。

DTPの理想

バッチ方式のTeXには問題点があります。それは，冒頭でも述べたように，文書が完成するまでどのような印字結果になるのかわからないという点です。文章や数式を書いている分にはいいのですが，表を作ったり図1のような図を描いたりするのはTeXがもっとも苦手とする分野です。

一方，WYSIWYGのDTPソフトにも問題点があります。それは，見たものしか得られないという点です。TeXが得意とする数式の処理を行おうとすると，自分でそれをレイアウトして画面に表示しなければなりません。画面に表示できなければ得ることはできないのです。

気分よく使えるDTPソフトとはどんなものなのか。数式を自分で組版したくはないし，手探りで図版を作りたくもない。結局，私の理想とするDTPソフトは，画面の3/4くらいにWYSIWYGで文書が表示されていて，1/4くらいにバッチ方式の文書が表示されている。どちらにも修正や変更を加えることができ，その結果は他方に反映されるといった，両者のインタラクティブな融合なのではないかと思います。スタイルファイルを取り入れたWYSIWYGのDTPソフトは，かなりこの線に近くなってきているような気がします。いずれにしても，これがかなり重い処理になることは間違いなく，やっぱり実用になるのはX68030やX88000（?）が登場してからになってしまうのかもしれません。

現在X68000で利用できるもっとも高品質なDTPソフトであるTeX。メモリとハードディスクに余裕のある方は，一度挑戦してみませんか。

図8（縮小率60%）

ようやくのことで日本語 LaTeX をキヤノン BJ-10v(360dpi) 対応でインストールし終わりました。

METAFONT の使い方もどうにか把握しました。

しかしながら，プリンタドライバ(print.x)は書き換えおよび再コンパイルの必要がありました。BJ-10vのコード体系は少し妙で，私が入手したバージョンのプリンタドライバとの相性がよくなかったのです。けっこう手間がかかりました。

やや副作用めきますが，改造する箇所を探してソースリストを眺めているうちに，\special 命令のハンドラを見つけました。title.sys フォーマットの画像ファイルを文書中に取り込むことができます。書式は次の通りです。

```
\special{!title title.sys}
\special{!titlerev title.sys}
```

後者は白黒を反転して取り込むものです。

使用上の注意としては，\special 命令は LaTeX とは無関係なので，絵を埋め込むためのスペースは自分で確保しなくてはならないということです。印刷したときの絵のサイズは 180dpi で換算してください(たとえば 180 ドットの絵なら 1 インチになります)。

title.sys フォーマットではいまいち使い勝手が悪いので，今回 CUT ファイルも取り込めるように拡張しました。書式は次の通りです。

```
\special{!cut sx.cut}
\special{!cutrev sx.cut}
```

360dpi の解像度は想像以上に素晴らしいようです。もちろん，ページプリンタに比べたら遅くてしかたがないのですが，この価格帯のプリンタのなかでは最高級の品質を持っているといえます。このくらい解像度がいいと、Z'sSTAFF PRO-68K のアウトラインフォントの欠点が目立ちさえしてしまいます。まだ第2水準の漢字も入れていません。入れると伏せ字になってしまいます(例:⊠珠)。

図9（縮小率60%）

What you see is ALL you get

〜TeXからのアプローチ〜

泉　大介

平成3年 6月 15日

少なからぬパーソナルコンピュータ上でDTPが進行しつつある状況を見ていると，標準添付のワープロが依然として主役の位置にある我らがX68000の環境はいささか貧弱であると言わざるをえません。Aldus の Pagemaker を筆頭とするDPTソフトの登場は，高品位な印刷物を自分で作成できる新しい環境を作り出しました。画面には印刷物のイメージが縮小して表示されており，それを見ながらタイトルをセットし，図版を貼り付けていく作業は，従来のワープロの世界とはまったく異なる環境を提供してくれます。

より細かい指定をしたい場合は画面表示を原寸大に，あるいは数倍にして作業することも可能になっています。タイトルや見出しを mm 単位で移動させるといった，まさに組版レベルのことが簡単に実行できるようになっている，しかも，その効果を目で確かめながら実行できるというのがこれらのDTPソフトの大きな特長です。目で見たものがそのまま得られるという意味で，この方式は WYSIWYG (What You See Is What You Get：ウィジウィグ) と呼ばれています。

TeX (テフまたはテックと読みます) はこの WYSIWYG 方式の対極に位置しています。作成する文書がどのようなイメージになるのかは，文書を印刷するまでわかりません。紙に印刷する代わりにディスプレイに印刷することもできますが，いずれにしても文書が完成するまでそのイメージを確認できない点は変わりません。これは TeX が，文字の大きさを何ポイントにするか，その行をセンタリングするのか右寄せにするのかなどといった情報を埋め込んだ文書ファイルを読み込み，一括して処理するバッチ方式を採用しているためです。

1　バッチ方式とTeX

普通のテキストファイル中に組版の情報を折り込むバッチ方式では，文書の作成から印刷までの手順は次のようになります(図1)。

1. まず〜.tex というファイル名で原稿を作成します。
2. それを TeX にかけると〜.dvi というファイルが作成されます。
3. もし，途中でエラーが出れば原稿を手直しして再び2.からやり直します。
4. めでたく〜.dvi ファイルが作成されたら，これをプレビューアと呼ばれるソフトにかけて画面に表示するか，プ

図10 原寸の出力例

今、拓かれるTEXの世界 BJ-10vがあなたをDTPへと誘う

1 フォントト自由自在

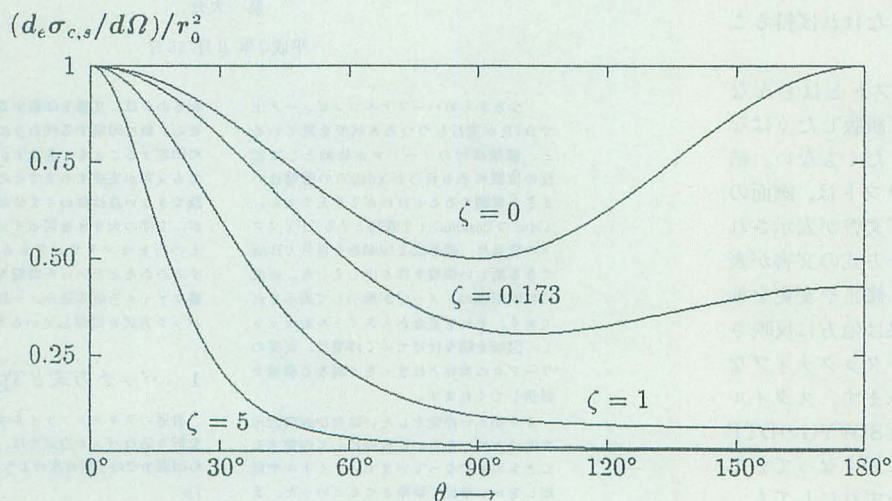
X68000用のTEXはZ's staffのベクタフォントを使用しています。Z's staffに付属しているものだけでなく，現在書体クラブとして「毛筆書体」「教科書体」などのフォントが発売されています。もちろん，この文書は明朝体で書いていますし，タイトルには「ゴシック」を使っています。次のセクションは試しに教科書体で作ってみました。

2 グラフ自由自在

丹氏の尽力によりTEXでCUTファイルが扱えるようになりました。そこで先月号のBASIC特集で掲載された、グラフィックをCUTファイルにするプログラムを使って早速実験をしてみることにしました。

実験に使ったのは、

$$\frac{d_e\sigma_{c.s}}{d\Omega}=\frac{r_0^2}{2}\left\{\frac{1}{1+\zeta(1-\cos\theta)}\right\}^3\left\{1+\cos^2\theta+\frac{\zeta^2(1-\cos\theta)^2}{1+\zeta(1-cos\theta)}\right\}\quad(cm^2/\text{電子})\qquad(1)$$

で与えられるコンプトン錯乱γ線のエネルギー角度分布です。このグラフは以下のようになります。

どうでしょうか。このようなグラフを取り込めると、ドキュメントの説得力もぐっとアップしますね。残念なのは解像度が180dpiになってしまう点ですが、これを360dpiにすると大きな図版を作りづらいという問題もあり難しいところです。このサンプルではグラフとX-Y軸をBASICで作り、文字はTEXで表示しています。

製品紹介

IOCS用FONT200書体

Kioi Makoto 紀尾井 誠

普段使っている文字を別の書体に切り替える，X68000ではそんなことも可能です。フォント切り替えに対応しているIOCS.Xを最大限に活用するための文字フォントが，なんと200書体まとめて発売されます。

古くはturbo console，そしてIOCS.XによってX68000の標準システムでも文字フォントの切り替えができるようになりました。ただ，肝心のフォントデータがシステムディスクその他には付属していないので，こういった機能を知らない人もいるのではないかと思います。

turbo console時代から，PDD(パブリックドメインデータ)では多くのフォントファイルが流通しています。なかでも，タイポグラファ平木敬太郎氏の作品は質量ともに他を圧倒しています。電脳倶楽部の購読者の皆さんにはお馴染みでしょうし，Oh! X1991年1月号の謹賀新年PRO-68Kにも氏のフォントの一部が掲載されています。

このたび，平木氏のオリジナルフォントがまとめてTAKERUで販売されることになりました。それも，どんと200書体です。

なお，これらの文字フォントをシステムに組み込んで使用する際にはIOCS.XをCONFIG.SYSで組み込んでおく必要があります。コマンドラインから実行して組み込んだ場合ではフォント切り替えの機能は使用できませんので注意してください。

切り替えは，

A>COPY ~.FON @IOCS

のように@IOCSという名前のファイルへフォントファイルをコピー動作することで行われます。

多彩なフォント

出力サンプルを200書体分，でーんと載せてみました(40%に縮小)。例文はあちこちで見掛けることもあるでしょう。どうして，茶色の狐が犬を跳び越えなければならないかというと，この文にはAからZまでの26文字がすべて含まれているからです。日本語の「いろは」に比べれば無駄がありますが，あちらの人もなかなかしゃれたことをやります。

つらつらと眺めていますと，一般的なゴシックやローマン体を初め，オカルティックなプログラミングには欠かせない鏡文字やあっという間の暗号文字(IBMと打つとHALになる)までサポートされています。主な書体には太さの違うものやさまざまなバリエーションが加わっています。

欠点は日常的に使えるフォントが少ないということでしょう。一度フォントを切り替えたあとはすべての文字が変わるので，変な書体にしてしまったらコマンドシェルの操作にも不都合が現れてきます。どちらかといえばワンポイントとして使えたら楽しいな，という文字が多いのです。テキストファイルの途中で文字の色を変えるように，フォントも自由に選択できたらと思うのは私だけではないでしょう。

IOCS.Xをマルチフォント対応に拡張すればすべては丸く収まるのですが。ユーザーが勝手にエスケープシーケンスの拡張を行うのもなんなので，メーカーの努力に期待したいところです。

使ってみる

さて，絵に描いた餅ではもったいないので1ページだけの簡単なマルチフォントツールを作ってみました。エスケープキーのあとにフォント名を入力するとフォントを切り替えます。あとは文字を入力するだけです。カーソルキーとコントロールキー(一部使えない)で簡単な編集もできます。プリンタへの出力はCOPYキーによるハードコピーを使ってください。

かなり強引なことをしていますが，自作プログラムからこれらのフォントを使用するのはそう困難ではないことがおわかりでしょう。とにかく，これだけのデータを活用しない手はありません。

IOCS用フォント：200書体　2,000円(税込)
ブラザー工業 (TAKERU) ☎052(824)2493

リスト1

```
 10 int x,y
 20 str a
 30 console 0,32,0
 40 locate 25 ,0
 50 repeat
 60   a=inkey$
 70   x=pos:y=csrlin
 80   if asc(a)=27 then fontchange()
 90   if asc(a)=8  then locate x-1,y:print" ";:locate x-1,y
100   print a;
110   y=csrlin
120   if asc(a)=13 then locate 25,y+1
130 until asc(a)=3
140 end
150 func fontchange()
160   int i,j
170   str n
180   char f(2047)
190   locate 0,30
200   input n
210   i=fopen("f:¥font¥"+n+".fon","r")
220   j=fopen("@iocs","w")
230   fread(f,2048,i)
240   fwrite(f,2048,j)
250   fclose(i)
260   fclose(j)
270   locate 0,30:print"                                 "
280   locate x,y
290 endfunc
```

表1 フォント名

1: 3_4	35: GOTHICA	69: MIRROR	103: PIZZAT	137: SLIMB	171: STOPS
2: 4_6	36: GOTHICB	70: NAMI	104: POMP	138: SLIMC	172: STOPV
3: 8_8	37: GOTHICC	71: NANAM	105: POMPA	139: SLIMJ	173: STORM
4: ALABIC	38: GOTHICCS	72: NUMA	106: POMPB	140: SLIMS	174: TATE
5: ANGO1	39: GOTHICE	73: NUMAB	107: POMPC	141: SLIMT	175: TATE2
6: ANGO2	40: GOTHICF	74: NUMAC	108: POMPD	142: SLOW	176: TATEC
7: ARR	41: GOTHICI	75: NUMAJ	109: POMPE	143: SMOKE	177: TATED
8: BISCUIT	42: GOTHICO	76: NUMAQ	110: POMPH	144: SMOKECS	178: TATEW
9: BOTTLE	43: GOTHICP	77: NUMAS	111: POMPJ	145: SMOKEI	179: TUBE
10: BUBLE1	44: GOTHICQ	78: NUMAT	112: POMPO	146: SMOKES	180: UNDC
11: CORN	45: GOTHICS	79: OLD	113: POMPP	147: SMOKET	181: UPL
12: EGW	46: GOTHICT	80: OLD2	114: POMPQ	148: SMOKEV	182: UU
13: EGYPT	47: GOTHICWB	81: OLDI	115: POMPS	149: SMOKEWS	183: VOLTA
14: EGYPT12	48: GOTHICWS	82: OLDL	116: POMPT	150: SNOW	184: WIDE
15: EGYPT2	49: GOTHICX	83: OPENG	117: POMPU	151: SNOW2	185: WIDE2
16: EGYPTB	50: GOTHICY	84: OPENGH	118: POMPW	152: SOFT	186: WIDE3
17: EGYPTC	51: GREEK	85: OPENGQ	119: POMPX	153: SOFTA	187: WIDEC
18: EGYPTCS	52: HEL1	86: PAN	120: PUF	154: SOFTB	188: WIDECS
19: EGYPTQ	53: KANA1	87: PIE	121: ROMAN1	155: SOFTC	189: WIDES
20: EGYPTS	54: KUMO1	88: PIE2	122: ROMAN2	156: SOFTD	190: WIDEW
21: EGYPTT	55: KUMO2	89: PIES	123: ROMAN3	157: SOFTE	191: WIND
22: EGYPTV	56: KUMO3	90: PIESS	124: ROMAN3CS	158: SOFTH	192: WIND2
23: EGYPTW	57: KUMOA	91: PIEV	125: ROMAN3S	159: SOFTO	193: WIND3
24: EGYPTWS	58: KUMOB	92: PIEW	126: ROMAN3T	160: SOFTP	194: WIND4
25: EGYPTZ	59: KUMOC	93: PIEWS	127: ROMAN3WS	161: SOFTQ	195: WINDCS
26: ELLEAIR	60: KUMOE	94: PIZZA	128: ROMAN4	162: SOFTS	196: WINDS
27: FTAIL1	61: KUMOO	95: PIZZAA	129: ROMAN4CS	163: SOFTT	197: WINDSS
28: FTAIL1B	62: KUMOP	96: PIZZAB	130: ROMAN4T	164: SOFTW	198: WINDV
29: FTAIL1C	63: KUMOQ	97: PIZZAC	131: ROMANQ	165: SPACE	199: WINDW
30: FTAIL1Q	64: KUMOS	98: PIZZAE	132: RUSSIAN	166: STONE	200: ZATU1
31: FTAIL1T	65: KUMOT	99: PIZZAO	133: SAKASA	167: STONEC	
32: GOTHIC	66: KUMOV	100: PIZZAP	134: SCRIPT	168: STONECS	
33: GOTHIC12	67: LEFT	101: PIZZAQ	135: SEN	169: STONES	
34: GOTHIC6	68: LINE	102: PIZZAS	136: SLIM	170: STONEV	

表2 フォント一覧

1 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
2 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
3 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog
4 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, [illegible]
5 2 PTHBJ AQNUM ENW ITLOR NUDQ SGD KZYX CNF, z pthbj aqnum enw itlor nudq sgd kzyx cnf.
6 0 QUECK BRAUN FAX JUMPS AVIR THI LOZY DAG, o queck brawn fax jumps avir thi lozy dag.
7 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
8 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
9 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
10 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
11 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
12 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
13 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
14 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
15 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
16 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
17 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
18 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
19 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
20 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.
21 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
22 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
23 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
24 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
25 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
26 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
27 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
28 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
29 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
30 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
31 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
32 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
33 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
34 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
35 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
36 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
37 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
38 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
39 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
40 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
41 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, *a quick brown fox jumps over the lazy dog.*
42 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
43 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
44 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
45 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.
46 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
47 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
48 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
49 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
50 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
51 Α ΘΤΙΕΚ ΒΡΟΩΝ ΦΟΧ ,ΤΜΠΣ ΟΣΕΡ ΤΗΕ ΛΑΖΨ ΔΟΓ, α θυιξκ βροων φοχ .υμπσ ο ερ τηε λαζψ δογ.
52 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
53 [illegible]
54 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
55 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
56 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
57 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
58 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
59 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
60 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
61 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
62 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
63 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
64 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.
65 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
66 **A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.**
67 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
68 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
.god yzal eht revo spmuj xof nworb kciuq a ,GOD YZAL EHT REVO SPMUJ XOF NWORB KCIUQ A 69
70 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.
71 *A quick brown fox jumps over the lazy dog, a quick brown fox jumps over the lazy dog.*
72 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

73 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

74 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

75 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

76 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

77 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.

78 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

79 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

80 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

81 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

82 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

83 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

84 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

85 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

86 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

87 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

88 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

89 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.

90 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

91 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

92 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

93 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

94 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

95 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

96 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

97 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

98 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

99 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

100 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

101 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

102 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.

103 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

104 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

105 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

106 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

107 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

108 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

109 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

110 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

111 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

112 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

113 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

114 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

115 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.

116 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

117 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

118 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

119 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

120 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

121 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

122 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

123 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

124 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

125 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.

126 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

127 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

128 *A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.*

129 *A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.*

130 *A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.*

131 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

132 Ф ЙГШСЛ ИКЩПТ АЩЧ ОГЬЗЫ ШМУК ЕРУ ДФЯН ВШП6 ф йгшсл икщпт ащч огьзы шмук еру дфян вшп6

133 (upside-down) A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

134 *A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.*

135 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

136 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

137 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

138 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

139 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

140 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.

141 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

142 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

143 *A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.*

144 *A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.*

145 *A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.*

146 *A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.*

147 *A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.*

148 *A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.*

149 *A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.*

150 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

151 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

152 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

153 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

154 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

155 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

156 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

157 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

158 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

159 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

160 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

161 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

162 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.

163 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

164 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

165 A_QUICK_BROWN_FOX_JUMPS_OVER_THE_LAZY_DOG,_a_quick_brown_fox_jumps_over_the_lazy_dog.

166 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

167 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

168 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

169 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.

170 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

171 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.

172 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.

173 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

179 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

180 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

181 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

182 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

183 A QÜICK BROWN FOX JÜMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

184 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

185 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

186 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

187 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

188 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

189 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.

190 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

191 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

192 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

193 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

194 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

195 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

196 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, A QUICK BROWN FOX JUMPS OVER THE LAZY DOG.

197 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

198 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

199 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

200 A QUICK BROWN FOX JUMPS OVER THE LAZY DOG, a quick brown fox jumps over the lazy dog.

ようこそここへC言語

標準入出力って何だろう

［第10回］

Nakamori Akira
中森 章

入出力装置に応じていちいちプログラムを書き換えるという無駄な作業は，できることなら避けたいもの。ということで，今月は標準入出力の登場です。さて，標準入出力とはいったいどんなものなのか，中森氏が懇切丁寧に解説します。

「クイズ宿題を忘れました」ではプレイヤーが小中学生の頃の問題が出題されます。しかし，問題が昔すぎてあまり答えられないので，生まれた年の入力は昭和47年以降（何歳サバを読んでるんだ）にしてしまう中森章です。それでも，正解率が70％にも満たない。うーん。

さて，今回のテーマは標準入出力です。これは入出力を扱うプログラムを書く場合はぜひとも知っておかなければならない概念です。この標準入出力を利用したプログラムの書き方を学ぶことにしましょう。

標準入力と標準出力とは

標準入力と標準出力。ただ聞いただけではなんのことやらわからない言葉ですね。意味は，字句のとおり，標準の入力および出力ということです。私たちは多くのプログラムが端末（キーボード）からデータを受け取り，端末（画面）に結果を出力することを知っています。ですから，端末が標準の入出力であると考えてもかまいません。しかし，標準という言葉は「端末」というひと言では片づけられない，重要な意味を持っているのです。

入力があって，それを処理し，結果を出力する。これがプログラムというものの基本的な構成でしたね。入力装置には端末，ファイル，RS-232Cのようなシリアル回線が考えられます。また，出力装置には入力と同様な端末，ファイル，シリアル回線のほかにプリンタやOPMなどのデバイスを考えることもできます。このように入出力装置にはいろいろの種類がありますが，入出力装置の種類によってプログラムを書き分けていたのでは大変です。ただ入出力装置が異なる程度なら単一のプログラムで処理をしたいものです。そこで，すべての入出力装置を代表する究極の入出力が考案されているのです。それが標準入力と標準出力です。

つまり，すべてのプログラムを，とりあえず，標準入力から流れてくるデータを読み込み，処理したデータを標準出力に書き出すというイメージで書いてしまうのです。プログラムをこのような構造にしておけば，あとから標準入出力の実体を切り替えることですべての入出力装置に対応することが可能になります[1]。標準入出力の初期値には，最もよく利用されるという理由で，端末が割り当てられていることが多いようです。

ところで，標準入出力とはいったんプログラムを作ったあと（コンパイルしたあと）から入出力を切り替えるものですから，BASICなどのインタプリタ言語しか知らない人にとっては画期的なことかもしれませんね。実際，X-BASICでは端末に対する入出力とファイルに対する入出力は命名によってしっかりと区別されていて，端末に対する入出力をファイルに切り替えるという芸当は（たとえXCでコンパイルしたあとでも）行えません（ファイル名にconを指定して端末をファイルと同等に扱うことはできるけど）。もし，標準入出力に馴染みがないようならこの機会にしっかりと覚えてしまいましょう。

1）この標準入出力という概念は目新しいものではない。早い時期から大型計算機のプログラミング言語では当たり前のように使用されてきた。FORTRANの参考書で，入力装置の番号が5，出力装置の番号が6となっているものがある。なぜ，この番号が1とか2でないか疑問を持った人もいるはず（FORTRANの参考書なんか見たことがない人のほうが多いかもしれないが）。実はこの意味ありげな5とか6という番号こそが標準入出力を示しているのだ。

標準入出力の切り替え

標準入出力をどのようにしてファイルなど端末以外の入出力装置に切り替えるかはOSに依存する機能です。これはC言語の文法とは直接は関係ありませんが，プログラムを実行できなければ意味がないので，ここで説明しておきましょう。Human68kでは，標準入出力を切り替える手段としてリダイレクトとパイプという機能が用意されています。これはワークステーションのOSであるUNIXを真似て取り入れられた機能です。

リダイレクトとは「再び（リ）指図する（ダイレクト）」という意味です。つまり，端末に割り当てられていた標準入出力を再び別の入出力装置に割り当てるように指図するということです。Human68kやUNIXでは，リダイレクトには不等号の＜と＞という記号を使います。＜が標準入力の，＞が標準出力のリダイレクトを示します。いま，標準入力からデータを読んで標準出力に結果を書くprog.xというファイルがあるとします。通常に，

prog

だけを実行したのでは，標準入力が端末（キーボード）に割り当てられているため，入力待ちの状態で停止しています。いつ入力待ちが終了するかはプログラムの書き方によってまちまちですが，一般には必要個数だけのデータを受け取ったとき，あるいはエンド・オブ・ファイル（CTRL-Zのコード）を受け取ったときに入力待ちが終了し，そして処理結果が端末（画面）に出てくるはずです。このとき，リダイレクト機能によって標準入力をinfileという名前のファイルに切り替えるのであれば，

prog < infile

を実行します。また，標準出力をoutfileという名前のファイルに切り替えるのであれば，

prog > outfile

を実行します。標準入力と標準出力を同時に切り替える，

prog < infile > outfile

というようなリダイレクトも可能です。

ところで，私たちはときどき，

prog >> outfile

のように標準出力のリダイレクトを>>によって行う場面を見ることがあります。これはprogが標準出力に書き込むデータをリダイレクト先のoutfileにアペンド（終わりに追加する）することを意味しています[2]。

次はパイプです。これは水道管を想像してもらえばよいでしょう。Human68kやUNIXでは，水道管が水をつぎつぎと引き渡していくように，標準入出力を流れるデータをパイプによって引き渡していくことができます。つまり，パイプとはプログラムが標準出力に書き込んだ結果を別のプログラムの標準入力につなぐ機能なのです。パイプは｜という記号で指定します。いま，prog1.xが標準出力に結果を書くプログラム，prog2.xが標準入力からデータを読み込むプログラムとしましょう。このとき，

prog1 | prog2

を実行すると，中間ファイルを作らなくてもprog1.xの出力結果をprog2.xの入力につなぎ替えることができるのです[3]。ここに標準入出力のもうひとつの意義が明らかになってきます。つまり，標準入力から読んで標準出力に書くプログラムをいくつかパイプでつなぎ合わせることで，複雑な処理を一括して行うことができるようになるのです。たとえば，

prog1 | find /n int | sort | more

というコマンドは，prog1.xが標準出力に書き出す結果から，intという文字列がある行だけを抜き出し（find.x），その結果をソートし（sort.x），その結果を1画面ずつ見る（more.x）という処理を行います（find.x，sort.x，more.xはHuman68kのコマンドですから説明はいりませんね）。この場合，文字列を検索しソートするというような2つの機能を持つプログラムを新たに作る必要はありません。あり合わせのプログラムをつなぎ合わせることで簡単に用を足してしまえるのですから。これがパイプの隠れた（？）利点なのです[4]。Human68kもそうですが，特にUNIXのツールでは，キーワードのある行を抜き出すとか，ソートするとか，同一行を削除するとか，大文字を小文字に変換するとかいった基本的な処理をするものばかりです。ただし，どれも標準入力から読んで標準出力に書くようになっているので，パイプでつないでやれば大きな威力を発揮させることができるのです。プログラムが特定のファイルに対して入出力を行うようになっていたのではこうはいきませんね。

ところで，Human68kの標準入出力はテキストファイルの入出力を対象としています。つまり，エンド・オブ・ファイル（CTRL-Zコード）を読み込んだ時点で標準入力が尽きたものと思ってしまうのです。つまり，標準入力をただ標準出力に書き出すだけのプログラムを作り，リダイレクトによってファイルのコピーを行うと失敗するので注意しましょう。もともとC言語で書く入出力を扱うプログラムはテキストファイルを変換するものがほとんどです。ですからテキストファイルだけしか処理できないとしても標準入出力の有用性が失われるということはないでしょう。

2）　UNIXには標準入力の切り替えのために<<という記号も用意されている。Human68kにはない。これは標準入力を端末などから一括して与えるときに使用する。<<の右に書かれた記号（文字列）と同じ記号が入力されるまでのデータをひとつのファイルとして標準入力に与える。たとえば，

```
prog << INPUT
1234
5678
INPUT
```

のように使用する。<<の右の記号を'（シングルクォート）で囲むと意味に若干の違いがあるがここでは深入りしない。興味ある人はUNIXの参考書を参照すること。

3）Human68kの場合，実際には中間ファイルが作られている。ただし，この中間ファイルは用がすむと自動的に消去されるので，ユーザーの目に触れることはない。

4）Human68kではそうともいえない。UNIXと異なり，Human68kのパイプ処理とは，いったん標準出力を中間ファイルに書き出したあと，次の標準入力に入力するものである。パイプを使用せずひとつのプログラムで複数の処理を実現する場合は，処理結果を中間ファイルを介さず，メモリ上で引き渡せるので高速な処理が期待できる。

標準入出力を扱う関数

標準入出力がだいたいわかったところでC言語で標準入出力を扱う主な関数を説明しましょう。scanf，printf，getsというこれまで何も考えず使用してきた関数もありますが，この機会に正確な使用法を説明しておきます。

●標準入力を扱う関数

1）　scanf

scanf関数は書式付きの入力変換を行います。すなわち，第1引数で指定される書式文字列にしたがって標準入力から入力されるデータ（一般にはテキストデータ）を変換し，第2引数以降に示される変数に代入します。

各引数はポインタ[5]でなければなりません。

scanf関数の実行は書式が尽きた(書式に合致するデータが取り出せた)とき，変換時にエラーが発生したとき，あるいは入力が尽きたときに終了します。変換が正常に終了すると第2引数以降の変数に代入したデータの数が戻り値となります。また，変換の途中にエラーが発生するか入力が尽きる場合はEOF(－1というint型データ)が戻り値となります。ただし，scanf関数が標準入力から読み込むデータの数は書式文字列に明示してあるのでその戻り値を参照するということはまずないでしょう。書式文字列には，標準入力を流れてくるテキストデータをどのように解釈し変換するかという変換仕様を記述してあります。変換仕様は%で始まる次の形式をしています。

% [*] [length] type

このうち [ ] で囲んだ部分は省略可能です。それぞれの記号の意味は次のようになっています。

* これは標準入力の中の対応するデータを読み飛ばすことを意味します。当然，その変換した値が変数に代入されることはありません(変数をscanfの引数に与えてはいけません)。

length これは標準入力から読み込むデータの最大幅を指定します。最大幅を超えたデータは切り捨てられます。標準入力のなかのデータが最大幅よりも小さい幅で表現されている場合は，この最大幅は意味を持ちません。

type これは変換文字です。10進数がd，16進数がxというように変数に取り込むデータ型によって指定する文字が決められています。変換文字の一覧を表1に示します。

表1 scanf関数の変換文字の一覧

変換文字	入力の型	変数（引数）の型
d, ld	10進整数	int型またはlong int型を指すポインタ
hd	10進整数	short int型を指すポインタ
i, li	2進，8進，16進，10進の整数	int型またはlong int型を指すポインタ
hi	2進，8進，16進，10進の整数	short int型を指すポインタ
b, lb	2進整数	int型またはlong int型を指すポインタ
hb	2進整数	short int型を指すポインタ
o, lo	8進整数	int型またはlong int型を指すポインタ
ho	8進整数	short int型を指すポインタ
x, lx	16進整数	int型またはlong int型を指すポインタ
hx	16進整数	short int型を指すポインタ
u, lu	符号なし10進整数	unsigned int型またはunsigned long int型を指すポインタ
hu	符号なし10進整数	unsigned short int型を指すポインタ
e, f, g	浮動小数点数	float型を指すポインタ
le,lf,lg	浮動小数点数	double型を指すポインタ
Le,Lf,Lg	浮動小数点数	long double型を指すポインタ
c	文字（空白文字を読み飛ばさない）	幅で指定された長さの文字を格納できる大きさを持つchar型配列へのポインタ
s	（空白文字を含まない）文字列	入力文字列と終了コード（¥0）を格納できる大きさを持つchar型配列へのポインタ
p	アドレスを表す16進数	ポインタを指すポインタ
n	読み込みはないこれまでの文字数	int型を指すポインタ
%	%という1文字	変数への代入は行われない
[...]	[ ] 内の各文字から成り立つ文字列	入力文字列と終了コード（¥0）を格納できる大きさを持つchar型配列へのポインタ
[^...]	[^] 内の各文字以外から成り立つ文字列	入力文字列と終了コード（¥0）を格納できる大きさを持つchar型配列へのポインタ

(注)XCVer.1ではlong double型はない。宣言してもdouble型とみなされて処理される。XCのver.2.0やGCCではlong double型はdouble型と同じ大きさを持つ。したがって，XCやGCCのscanfではle,lf,lgとLe,Lf,Lgは同一である。
(注) XCVer.1のscanfでは変換文字としてi,b,p,nはサポートしていない。

この変換仕様のほかにも書式文字列の中には空白文字(空白，タブ¥t，改行¥n，垂直タブ¥v，改ページ¥f，復帰¥r)と通常文字を書くことができます。空白文字は特に意味はなく読み飛ばされます。通常文字はそれに合致する文字が標準入力にある場合のみ，標準入力からの読み込みを続けるような指定をするためのものです。

ところで，scanf関数の引数の数は可変になっています。scanf関数は与えられた引数の数を書式文字列の中の%の数によって認識しようとしますから，第2引数以降の変数の数は書式文字列の中の%の数と同じ（か，それ以上)でなければなりません。もし，そうでなければ(%の数のほうが多い場合）プログラムが誤動作する恐れがありますから注意しましょう。なお，scanf関数の書式文字列の使用例は，今回の「基礎力を高めよう」で取り上げていますのでそちらを参考にしてください。

2） gets

gets関数は標準入力からテキストデータを1行ずつ読み込むときに使用します。標準入力からの1行分（改行¥nに達するまで）の文字列を引数で与えられるchar型の配列に読み込みます。読み込まれた改行文字（¥n）は文字列の終了コード(¥0)に変換されます。gets関数の引数としては（正確には）配列を指すポインタを与えますが，1次元配列の場合は配列名を渡しておけばよいでしょう。戻り値は引数として与える配列を指すポインタですが，エラーが発生したり入力が尽きた場合にはNULL（値0）が返ります。

3） getchar

getchar関数は標準入力から1文字ずつ読み込むときに使用します。標準入力からの1文字をint型の符号なし文字に変換して戻り値とします。もし，エラーが発生したり入力が尽きた場合にはEOF（値－1）が返ります。

●標準出力を扱う関数

1） printf

printf関数は書式付きの出力変換を行います。すなわち，第1引数で指定される書式文字列にしたがって第2引数以降で与えられるデータを変換して文字列を作り，標準出力に書き込みます。printf関数の戻り値は標準出力に書き込んだ文字列の長さです。もし，エラーが発生した場合は負の数が返ります[6]。ただし，printf関数の戻り値を何に使用するのかは定かではありません。

書式文字列には標準出力に書き込む普通の文字列と%で始まる変換仕様を記述してあります。そして，書式文字列の中の変換仕様の部分に，その仕様にしたがって引数を変換した文字列が埋め込まれて，最終的に標準出力

に書き込む文字列が出来上がります。変換仕様は%で始まる次の形式をしています。

% [flag] [length] [.digit] type

このうち [ ] で囲んだ部分は省略可能です。それぞれの記号の意味は次のようになっています[7]。

flag　これは変換仕様を補足するための指定です。flagの値と内容は次のとおりです。またflagは順不同で複数個指定ができます。

- －　変換された引数を左詰めで印字することを指定。
- ＋　数値を＋または－の符号付きで印字することを指定。
- 空白　数値が正数のときは空白を，負数のときは－の符号付きで印字することを指定。＋と同時に指定されると無効。
- 0　数値変換のとき印字幅一杯に0を左詰めして印字することを指定。－と同時に指定されると無効。
- #　別の出力形式を指定。typeによって変換の行われ方が異なる。詳しくはXCのマニュアルを参照のこと。

length　これは引数を変換するときの最小文字数を10進数で指定します。変換した文字数がlengthよりも少ないときは，文字の左側を空白で埋めて右揃え（flagに－が指定されたときは逆）で印字されます。実際に変換した結果がlengthよりも大きくなる場合，あるいはlengthが指定されてない場合は，すべての文字を印字します。なお，flagに0が指定されている場合は，空白の代わりに0で文字が埋められます。

lengthを10進数ではなく＊という記号で指定した場合は，lengthをprintf関数の引数で指定することができます。

digit　これは実際に印字する最大文字数，あるいは浮動小数点データを変換するときの小数点以下の桁数を10進数で指定します。変換した文字数，あるいは小数点以下の桁数がdigitよりも小さいときは0が埋められて印字されます。

digitを10進数ではなく＊という記号で指定した場合は，digitをprintf関数の引数で指定することができます。

type　これは変換文字です。printf関数への書式文字列以外の引数はint型かdouble型，あるいはchar型へのポインタで与えられてきますが，それをどのようなデータとして解釈して印字すべきかを指定します。変換文字の一覧を表2に示します。

ところで，scanf関数と同様にprintf関数の引数の数も可変になっています。printf関数も与えられた引数の数を書式文字列の中の%（と＊）の数によって認識しようとしますから，第2引数以降の変数の数は書式文字列の中の%の数と同じ（か，それ以上）でなければなりません。もし，そうでなければ（%の数のほうが多い場合）不定な値を標準出力に書き出してしまうだけでなく，運が悪ければプログラムが誤動作することがありますから注意しましょう。なお，printf関数の書式文字列の使用例は今回の「基礎力を高めよう」で取り上げていますのでそちらを参考にしてください。

2）putchar

putchar関数は標準出力に1文字ずつデータを書き込むときに使用します。通常は負でない値を戻り値としますが，エラーが発生した場合はEOF(－1)が返ります。

5）ポインタについては次回以降で説明する。とりあえず，書式指定は"（ダブルクォート）で囲まれた文字列，第2引数以降は変数名の頭に＆（アンドあるいはアンパサンド）を付けたもの（1次元配列ならば配列名そのもの）と思っておいてよい。
6）処理系によっては違う値が戻り値となることがある。printf関数の戻り値を使うプログラムを移植するときには注意を要する。
7）flagの0の扱いはXCでは特殊である。lengthを指定するときの10進数の前に書かなければ認識されない。したがって，＊を使用してlengthの値を入力するとき，文字の左側を0で埋めるというような指定はできない。

標準入出力を用いるプログラム

ご承知のようにC言語はワークステーションやパソコン上で使用することができます。ときには，パソコンで作成したプログラムをワークステーションに持ってくることもあればその逆もあるでしょう。このようなとき，うっかりするとパソコンとワークステーションにおける

表2　printf関数の変換文字の一覧

変換文字	引数の型	引数の解釈
d, ld i, li	int型	int型またはlong int型とみなして10進数で印字
hd hi	int型	short int型とみなして10進数で印字
b, lb	int型	unsigned int型とみなして符号なし2進数で印字
hb	int型	unsigned short int型とみなして符号なし2進数で印字
B,lB,hB	int型	b,lb,hbに同じ
o, lo	int型	unsigned int型とみなして符号なし8進数で印字
ho	int型	unsigned short int型とみなして符号なし8進数で印字
x, l x	int型	unsigned int型とみなして符号なし16進数で印字
hx	int型	unsigned short int型とみなして符号なし16進数で印字
X,lX,hX	int型	x,lx,hxに同じだが16進数のａ～ｆをＡ～Ｆとして印字
u, lu	int型	unsigned int型とみなして符号なし10進数で印字
hu	int型	unsigned short int型とみなして符号なし10進数で印字
f,lf	double型	[－]ddd.dddという形式の10進数浮動小数点数で印字
e,le	double型	[－]d.dd e [－]ddという形式の10進数浮動小数点数で印字
E,lE	double型	[－]d.dd E [－]ddという形式の10進数浮動小数点数で印字
g,lg	double型	ｆかｅのうち変換後の文字数が短いほうの形式で印字
G,lG	double型	ｆかＥのうち変換後の文字数が短いほうの形式で印字
c	int型	unsigned char型とみなして対応する文字を印字
s	char＊ 型	終了文字（＼0）がくるまで、あるいは指定した桁数が尽きるまで文字列を印字
p	void＊ 型	ポインタとみなして処理系依存の表現で印字
n	int＊ 型	これまでに書かれた文字数を印字 引数は参照しない

（注）XCVer.1のprintfでは変換文字としてi,b,p,nはサポートしていない。

テキストファイルの文字コードの扱いで不都合を生じることがあります。たとえば，UNIXが主流のワークステーションでは改行コードは0x0Aという1バイトですが，MS-DOSが主流のパソコンでは改行コードは0x0D，0x0Aの2バイトです。パソコン上のソースプログラムをワークステーションに持ってくる場合，0x0D, 0x0Aというコードがきたら0x0Aに変換する（0x0Dを無視する）処理を行わなければなりません（それ以外はどちらもASCIIコードなので漢字を使わなければ同一です）。多くの通信ソフトでは，テキストファイルに対してはその作業を自動的に行ってくれます。しかし，パソコン側でLHarc（X68000ではLH）などで圧縮結合したソースファイルを転送する場合には，テキストとして転送することができない（バイナリモードでファイルのイメージをそのまま転送する）ので，0x0D, 0x0Aを0x0Aに変換するという処理を行うことはできません。結果として，ワークステーションに持ってきたソースファイルには，余計な0x0Dのコードが残ってしまうことになります。この0x0Dというコードはコンパイラが認識できませんから当然コンパイルエラーになってしまいます。そんなとき，余計な0x0Dのコードをエディタなどでひとつずつ削除していってもいいのですが，C言語を知っている人ならば，0x0Dを削除するプログラムを書いてソースファイルを変換してしまおうと考えるでしょう[8)]。ちょっと慣れたプログラマなら条件反射的に，

```
main( )
{
    int ch;
    while((ch=getchar( )) != (-1)) {
        putchar(ch);
    }
}
```

という雛型のプログラム（これはEOFになるまで標準入力から読んだデータをそのまま標準出力に書いているだけ）を書き，putcharの前の行に0x0Dを無視するための，

```
if(ch=0x0d) continue;
```

を入れて目的のプログラムを完成させてしまうのです。この間，わずか1～2分といったところでしょうか。あとは，Cシェルのforeach（MS-DOSやHuman68kのforに相当する）を使って対象となるすべてのファイルを標準入力にリダイレクトしては0x0Dのコードを削除した結果を標準出力から取り出して元のファイルに書き戻すという処理を行えばよいのです。

ここで大事なことは，標準入力を標準出力に書き出すプログラムをすぐに書けるかどうかということです。これは経験を積むしかありません。しかし，このような状況は結構ありますから，起こりうる場合を想定してある程度のプログラムの雛型を覚えておくのは有用です。以下に，標準入出力を扱うプログラムの例をいくつかのタイプに分けて説明します。

●scanfによる入力

scanf関数は引数の書式文字列の中にデータの個数が記述されていることからもわかるように，プログラムで必要なデータの個数はあらかじめわかっています。scanf関数を使う入力は，処理に必要な数の入力データが揃うまでfor文なりwhile文でscanf関数を繰り返して呼び出すだけです。scanf関数を使用する場合の定型的なプログラムは特にありませんが，とりあえずリスト1にscanf関数を使ったプログラム例を示しておきましょう。リスト1は標準入力（ファイルをリダイレクトする）から入力される9個の浮動小数点データを3×3行列の各要素とみなして転置処理を行い，結果をprintf関数で標準出力に書き込むプログラムです。for文で合計9回scanf関数を呼び出しているのがわかりますね。

●1行単位の処理

C言語を使ったプログラムで最も多いのはテキストファイルを変換するプログラムです。これは標準入力から1行を読み込んではそれを処理し，その結果を標準出力に書き込んでいく処理が主流です。この場合は標準入力からのデータの受け取りにgets関数，処理結果の標準出力への書き込みにprintf関数を使用すればよいでしょう。そのプログラムのパターンは，

```
char LINE [1000] ; /*  1行分を記憶する配列 */
while ( gets ( LINE ) != NULL ) {
```

リスト1　scanf関数を使ったプログラム

```
 1: /*
 2:      標準入力からのデータを受け取るプログラム
 3:
 4:      (例) 9個の浮動小数点データを3×3行列の
 5:           要素とみなして転置する
 6: */
 7: double A[3][3];
 8:
 9: main()
10: {
11:      int i,j;
12:      for(i=0;i<3;i++)   /* 規定回数読む */
13:           for(j=0;j<3;j++)
14:                scanf("%lf",&A[i][j]);
15:      CONVERT();            /* 処理 */
16:      for(i=0;i<3;i++){ /* 結果を書く */
17:           for(j=0;j<3;j++)
18:                printf("%lf ",A[i][j]);
19:           printf("¥n");
20:      }
21: }
22:
23: CONVERT()
24: {
25:      int i,j;
26:      for(i=0;i<3;i++)
27:           for(j=i+1;j<3;j++){
28:                double tmp;
29:                tmp=A[i][j];
30:                A[i][j]=A[j][i];
31:                A[j][i]=tmp;
32:           }
33: }
```

リスト1の実行結果　a)入力　B)出力

```
a) 入力ファイル

  1.2 3.4 5.6
  6.7 7.8 9.1
  1.1 2.2 3.7

b) 出力

  1.200000 6.700000 1.100000
  3.400000 7.800000 2.200000
  5.600000 9.100000 3.700000
```

LINE［ ］の内容を適当に処理；

printf("%s¥n", LINE)；

}

でよいでしょう。標準入力から1行読んでは処理をし，標準出力へ書き出すという処理をデータが尽きるまで（gets関数がNULLを返すまで）while文で繰り返し行うだけです。特に問題はありませんね。NULLという記号は0と書いてもよいのですが，NULLを待っていることを明示するためにこう書いてあります。ただし，このNULLという定数はプログラムの中では定義されていませんから，それを定義するために，

#include <stdio.h>

という1行をプログラムの先頭に書いておきましょう。これは標準的な入出力に関係する定数やデータ型を定義するおまじないです[9]。

さて，標準入力からのデータを1行ずつ処理するプログラムの例をリスト2に示します。また，リスト2のプログラムの実行時に標準入力へリダイレクトするテスト用のファイルをリスト3に示してあります。リスト2のプログラムで入出力を行っている部分はパターンどおりですね。リスト2では，読んだ1行の処理（CONVERTという関数）として，大文字を小文字に，小文字を大文字に変換するというほとんど意味のない処理を行っていまこの変換処理も説明は不要でしょう。ただ，CONVERTという関数の中で行っている全角文字を無視する処理が気になる人がいるかもしれないので，それは説明しましょう。

試しに全角文字を無視する処理の部分を省略したプログラムをコンパイルして，リスト3の内容のファイルを標準入力にリダイレクトして実行してみてください。結果が思ったようになりませんね。全角文字で書いてある部分がおかしくなってしまうのがわかると思います。これは，全角文字は2バイトで1文字を表しているため，その2バイト目が運悪く，

if(ch>='a' && ch<='z')

あるいは，

if(ch>='A' && ch<='Z')

に当てはまるような文字コードであった場合にまったく別の全角文字に変換されてしまうためです。このことを避けるため，全角文字の1バイト目を読み込んだときには次の1文字を変換しないという処理をやっているのです。日本語を扱うプログラムの難しさがこんなところに現れているのですね。ところで，同じCONVERTという関数の中で，

ch&=0xff；/*Line［ ］がunsignedなら不要*/

という部分がありますが，これがなぜ必要かはわかりますね。ヒントは符号拡張です。

●ファイル全部を読んでからの処理

標準入力を1行ずつ読み込んで処理をするプログラムの変形として，標準入力を終わりまで読み込んでから，読み込んだファイル全体を処理し，そのあと標準出力に1行ずつ書き出していくというプログラムも考えられます。標準入力から入力されてくるテキストファイルの各行が関連性のないものであれば，1行単位で読み書きを行えばいいのですが，プログラムのソースファイルのよ

リスト2　1行単位に処理するプログラム

```
 1: /*
 2:         テキストファイルを変換するプログラム
 3:
 4:          (タイプ1：1行単位で処理)
 5:
 6:          (例) 大文字を小文字に，小文字を大文字に
 7: */
 8: #include <stdio.h>
 9:
10: char Line[256]; /* 1行の最大文字数は 256 バイト */
11:
12: main()
13: {
14:         while( gets(Line)!=NULL ){     /* 読む */
15:                 CONVERT();              /* 変換 */
16:                 printf("%s¥n",Line); /* 書く */
17:         }
18: }
19:
20: CONVERT()
21: {
22:         int i;
23:         int ch;
24:         for(i=0;i<256;i++){
25:                 ch=Line[i];
26:                 if(ch==NULL) break;
27:
28: /* 全角文字を無視する処理 */
29:                 ch&=0xff; /* Line[] が unsigned なら不要 */
30:                 if((ch>=0x80 && ch<0xa0)||(ch>=0xe0)){
31:                         i++;
32:                         continue;
33:                 }
34: /* ここまで */
35:
36:                 if(ch>='a' && ch<='z')
37:                         Line[i]=ch-'a'+'A';
38:                 else if(ch>='A' && ch<='Z')
39:                         Line[i]=ch-'A'+'a';
40:         }
41: }
```

リスト2の実行結果

```
/*
  テスト用の入力ファイル

 《全角文字のパターン》

    !"＃Ｓ％＆' ()＊＋，－．／０１２３４５６７８９：；＜＝＞？
  ＠ＡＢＣＤＥＦＧＨＩＪＫＬＭＮＯＰＱＲＳＴＵＶＷＸＹＺ［￥］＾＿
  'ａｂｃｄｅｆｇｈｉｊｋｌｍｎｏｐｑｒｓｔｕｖｗｘｙｚ｛｜｝～

  《半角文字のパターン》

   !"#$%&'()*+,-./0123456789:;<=>?
  @abcdefghijklmnopqrstuvwxyz[¥]^_
  `ABCDEFGHIJKLMNOPQRSTUVWXYZ{|}~

*/
MAIN()
{
    PRINTF("tHIS IS ONLY THE beginning.¥N");
}
```

リスト3　リダイレクト用の入力ファイル

```
 1: /*
 2:    テスト用の入力ファイル
 3:
 4:   《全角文字のパターン》
 5:
 6:     !"＃Ｓ％＆' ()＊＋，－．／０１２３４５６７８９：；＜＝＞？
 7:    ＠ＡＢＣＤＥＦＧＨＩＪＫＬＭＮＯＰＱＲＳＴＵＶＷＸＹＺ［￥］＾＿
 8:    'ａｂｃｄｅｆｇｈｉｊｋｌｍｎｏｐｑｒｓｔｕｖｗｘｙｚ｛｜｝～
 9:
10:    《半角文字のパターン》
11:
12:     !"#$%&'()*+,-./0123456789:;<=>?
13:    @ABCDEFGHIJKLMNOPQRSTUVWXYZ[¥]^_
14:    `abcdefghijklmnopqrstuvwxyz{|}~
15:
16: */
17: main()
18: {
19:     printf("This is only the BEGINNING.¥n");
20: }
```

うに前後の行がお互いに関連を持っているものが入力されてくる場合は，それをいったん全部引き取ってから処理を行う必要があります。アセンブラやコンパイラのプログラムを作るときもこのような場合に分類できるでしょう。そんな場合のプログラムのパターンを考えます。
標準入力から読み込んだデータを保存する場所として，1行単位の処理では1次元配列を考えました。今度は各行ごとに保存する必要がありますから，2次元配列が必要です。先の1行単位の処理でのデータの格納場所であるLINEという配列を2次元に拡張すれば，

```
char LINE [1000] [1000]；
```

という宣言になると思います。この2次元配列の宣言は，

```
char LINE [0] [1000]；/＊ 1行目を格納 ＊/
char LINE [1] [1000]；/＊ 2行目を格納 ＊/
              ⋮
char LINE [999] [1000]；/＊1000行目を格納 ＊/
```

という1000個の1次元配列を一括して宣言したものと考えることができます。これを眺めると，2次元配列の要素で右側の［ ］を取ったものが1次元配列でいうところの配列の名前に相当することがわかりますね。そこで，標準入力からの最初の1行を配列に取り込むためには，

リスト4　全部読んで処理するプログラム

```
 1: /*
 2:      テキストファイルを変換するプログラム
 3:
 4:      (タイプ2：ファイルを全部読んでから処理)
 5:
 6:      (例) 単純な行単位のソートプログラム
 7: */
 8: #include <stdio.h>
 9:
10: char Line[1000][256]; /* ファイルの最大行数は 1000 行 */
11:
12: int  maxLine; /* 行数を保持する変数 */
13:
14: main()
15: {
16:     int i;
17:
18:     maxLine=0;
19:
20:     while(gets(Line[maxLine])!=NULL) /* 読む */
21:         maxLine++;
22:     CONVERT();                       /* 変換 */
23:     for(i=0;i<maxLine;i++)           /* 書く */
24:         printf("%s¥n",Line[i]);
25: }
26: /*
27:     行のソートにはライブラリ関数を使っている
28:
29:      (これがメインではないのでちょっと手抜き)
30: */
31: CONVERT()
32: {
33:     int strcmp();
34:
35:     qsort(Line,maxLine,256,strcmp);
36: }
```

リスト4の実行結果

```
    printf("This is only the BEGINNING.¥n");
   !"#$%&'()*+,-./0123456789:;<=>?
  @ABCDEFGHIJKLMNOPQRSTUVWXYZ[¥]^_
   `abcdefghijklmnopqrstuvwxyz{|}‾
    ！”＃＄％＆’（）＊＋，－．／０１２３４５６７８９：；＜＝＞？
  《半角文字のパターン》
   ＠ＡＢＣＤＥＦＧＨＩＪＫＬＭＮＯＰＱＲＳＴＵＶＷＸＹＺ［￥］＾＿
   テスト用の入力ファイル
  ‘ａｂｃｄｅｆｇｈｉｊｋｌｍｎｏｐｑｒｓｔｕｖｗｘｙｚ｛｜｝～
  《全角文字のパターン》
*/
/*
main()
{
}
```

```
gets(LINE [0])
```

という引数で，その次の1行を配列に取り込むためには，

```
gets(LINE [1])
```

という引数でgets関数を呼び出せばよいことがわかります。結局，これからデータを読み込むべき行がmaxlineという変数で示されているのなら，

```
gets(LINE [maxline])
```

によってデータ（maxline+1行目のデータ）を配列に取り込めばよいことがわかります。このとき，

```
char LINE [1000] [1000]；
maxline＝0；
while(gets(LINE [maxline])!＝NULL) {
    maxline++；
}
 LINE [1000] [  ] の内容を適当に処理；
for(i＝0；i<maxline；i++) {
   printf("%s¥n",LINE [i])；
}
```

というのがプログラムのパターンになります。標準入力からデータを，それが尽きるまでLINEという2次元配列に読み込み，それを処理し，最後にまとめて標準出力に書き出すという，最初に示したscanf関数を使った例に似たパターンになっています。
リスト4に標準入力を最後まで読んで処理するプログラムの例を示します。これは標準入力からすべての行を読み込んだあと，XCのライブラリ関数のqsort（クイックソート）を使って行単位のソートを行い，標準出力に書き出すという処理を行います。Human68kのコマンドであるsort.xの最も単純なものと思ってくれたらよいでしょう。リスト4のプログラムの実行結果は，リスト3の内容のファイルを標準入力にリダイレクトした結果です。なお，ライブラリ関数のqsortの使用法はここでは説明を省きます。詳細を知りたい人はXCのライブラリマニュアルを参照してください。

●1文字単位の処理

標準入力からのデータを1行単位に処理するプログラムがあるならば，それを1文字単位で処理するプログラムがあってもおかしくありません。標準入力を1行単位に読み込んで処理するプログラムでは改行コード（¥n）までのデータをいったん別の保存場所に格納したあとで処理をしているのと同じことですから，1行単位の処理は1文字単位の処理に含まれると思ってよいのです。このような1文字単位の処理ではgets関数の代わりにgetchar関数を使うことになります。実際，gets関数はgetchar関数を使って，

```
gets(s)
char s [  ]；
{
    int i＝0；
```

```
    while(1) {
        s [i]=getchar( );
        if(s [i]==EOF) return(NULL) ;
        if(s [i]=='¥n') break ;
        i++ ;
    }
    s [i]=NULL ;
    return(s) ;
}
```

と書き下すことができます。またscanf関数も，ちょっと面倒臭そうですが，getchar関数を使って書き換えることができそうです。結局，getchar関数さえあれば，標準入力を扱うすべての場合に対応できてしまうのです。したがって，getchar関数は標準入力を扱う関数のなかで一番大事な忘れてはならない関数だということができます。

標準入力を1文字単位に処理する場合のプログラムのパターンは，

```
while((ch=getchar( )) !=EOF) {
    chに対して適当な処理；
    putchar(ch) ;
}
```

となります。標準入力から入力されるデータを，それが尽きるまで，1文字単位に読み込み，適当な処理をして，1文字単位に標準出力に書き出すという，これ以上くずしようのない単純なパターンですね。なおEOFという定数を使うためには，いつもどおりプログラムの先頭に，

```
#include <stdio.h>
```

の1行が必要になります。ところで，リスト2の1行単位のプログラムなどは1文字に対してのみ変換を行うものですから，こちらの1文字単位のパターンでプログラムを書いたほうがすっきりしたかもしれませんね。

1文字単位に標準入力の文字を扱うプログラムの例をリスト5に示します。これは，標準入力から入力される半角文字をそれに対応する全角文字に変換して標準出力に書き出すプログラムです。プログラムで行っていることは，入力された半角文字（全角文字はそのまま書き出している）を添字の値として，半角文字の文字コード順に全角文字が格納されている配列の中から2文字（全角文字1文字分）を取り出して標準出力に書いているだけです。配列を参照するときに（全角文字が2文字単位に格納されているので）添字を2倍することに注意すれば，難しいところはありませんね。リスト5の実行結果はリスト3の内容のファイルを標準入力にリダイレクトした結果です。

---

8)　エディタでの一括置換やAWKなどのユーティリティを利用することも考えられるが，この程度の処理ならC言語でプログラムを書いても手間に大差はない。

9)　いつもいつも「おまじない」で終わらせてしまうのは恐縮なので少し説明しておこう。#includeはCコンパイラのプリプロセッサの命令で，指定したファイルをその位置に取り込むことを示す。stdio.hというファイルには標準的な入出力に関連する定数やデータ型を定義してある。

◆基礎力を高めよう

設問1　次に示す文字列が標準入力から与えられたときに，scanf関数を実行したときに引数で示す変数に代入される値を答えてください。ただし，

```
int x, y, z;
char a [100],b [100] ;
```

という変数が宣言されているものとします。

1)標準入力　"123 456 789"

```
scanf("%d %*d %d",&x,&y,&z) ;
```

2)標準入力　"12345 5678 9048"

```
scanf("%2d %2d %2d",&x,&y,&z) ;
```

3)標準入力　"1234 0x1234 abcd"

```
scanf("%x %x %x",&x,&y,&z) ;
```

4)標準入力　"10 x 2y 3z"

```
scanf("%d x %d y %d",&x,&y,&z) ;
```

リスト5　1文字単位に処理するプログラム

```
 1: /*
 2:      テキストファイルを変換するプログラム
 3:
 4:      （タイプ3：1文字単位で処理）
 5:
 6:      （例）半角文字を全角文字に変換
 7: */
 8: #include <stdio.h>
 9:
10: char zen1[]="　！”＃＄％＆’（）＊＋，－．／¥
11: ０１２３４５６７８９：；＜＝＞？＠ＡＢＣＤＥ¥
12: ＦＧＨＩＪＫＬＭＮＯＰＱＲＳＴＵＶＷＸＹＺ［¥
13: ￥］＾＿‘ａｂｃｄｅｆｇｈｉｊｋｌｍｎｏｐｑ¥
14: ｒｓｔｕｖｗｘｙｚ｛｜｝～";
15:
16: char zen2[]="。「」、・ヲァィゥェォャュョッー¥
17: アイウエオカキクケコサシスセソタチツテトナニヌ¥
18: ネノハヒフヘホマミムメモヤユヨラリルレロワン゛゜";
19:
20: main()
21: {
22:     int ch;
23:
24:     while((ch=getchar())!=EOF){ /* 読む */
25:         /* 変換して書く */
26:         if((ch>=0x80 && ch<0xa0)||(ch>=0xe0)){
27:             putchar(ch);
28:             putchar(getchar());
29:         }
30:         else if(ch>=' ' && ch<='~'){
31:             putchar( zen1[2*(ch-' ')] );
32:             putchar( zen1[2*(ch-' ')+1] );
33:         }
34:         else if(ch>='｡' && ch<='ﾟ' ){
35:             putchar( zen2[2*(ch-'｡')] );
36:             putchar( zen2[2*(ch-'｡')+1] );
37:         }
38:         else
39:             putchar( ch );
40:     }
41: }
```

リスト5の実行結果

```
/*
 テスト用の入力ファイル
《全角文字のパターン》

 ！”＃＄％＆’（）＊＋，－．／０１２３４５６７８９：；＜＝＞？
＠ＡＢＣＤＥＦＧＨＩＪＫＬＭＮＯＰＱＲＳＴＵＶＷＸＹＺ［￥］＾＿
‘ａｂｃｄｅｆｇｈｉｊｋｌｍｎｏｐｑｒｓｔｕｖｗｘｙｚ｛｜｝～

《半角文字のパターン》

 ！”＃＄％＆’（）＊＋，－．／０１２３４５６７８９：；＜＝＞？
＠ＡＢＣＤＥＦＧＨＩＪＫＬＭＮＯＰＱＲＳＴＵＶＷＸＹＺ［￥］＾＿
‘ａｂｃｄｅｆｇｈｉｊｋｌｍｎｏｐｑｒｓｔｕｖｗｘｙｚ｛｜｝～

*/
main()
{
    printf("This is only the BEGINNING.¥n");
}
```

5)標準入力 ”The Computer Music”

scanf(”%4c%10s”,a,b)；

6)標準入力 ”The Computer Music”

scanf(”%2s %2s”,a,b)；

7)標準入力 ”a b b cea”

scanf(”% [abcd] ”,a)；

8)標準入力 ”new world”

scanf(”% [^ abcd] ”,a)；

設問 2　次に示すprintf関数の印字結果が同じものを指摘してください。

1) printf(”%+010d¥n”,123)；
2) printf(”%0+10d¥n”,123)；
3) printf(”%+ 10d¥n”,123)；
4) printf(”% +10d¥n”,123)；
5) printf(”%+010.d¥n”,123)；
6) printf(”%0+10.d¥n”,123)；
7) printf(”%+ 10.d¥n”,123)；
8) printf(”% +10.d¥n”,123)；
9) printf(”%+010.9d¥n”,123)；
10) printf(”%0+10.9d¥n”,123)；
11) printf(”%+ 10.9d¥n”,123)；
12) printf(”% +10.9d¥n”,123)；

設問 3　printf関数で出力幅 (length) を引数から指定する場合は,

printf(”%＊d¥n”,5,10)；

のように書きます。この例は5桁で10という値を右詰めで出力し，出力幅に満たない部分は空白で埋められます。ただし，XCでは出力幅を＊で指定したときには，出力幅に満たない部分を0で埋めることはできません。つまり，%0＊dという書式指定が許されません。出力幅を引数で指定しながら出力幅に満たない部分を0で埋めるための方法を考えてください。

おわりに

標準入出力が自由に使えるようになればC言語のプログラミングの第1段階は終了です。これまでに解説した部分だけで基本的なプログラムを書くために必要な知識はすでに備わっているはずですから，あとはいかにたくさんのプログラムを書いたかによってC言語が身につくか否かが決まってきます。K＆Rの第1章から第4章の練習問題などは手頃な題材ですから挑戦してみてはいかがでしょう。

さて，来月からは待ちに待った（？）ポインタの話題に入ります。ポインタでC言語に挫折する人が多いとよく聞きますが，この連載を読めばきっと（たぶん，おそらく）ポインタがわかるようになります（なるんじゃないかな）。

まあとにかく，一緒に勉強していきましょう。

◆基礎力を高めようの解答

設問 1

1)　x：123 y：789 z：不定
2)　x：12 y：34 z：5
3)　x：4660(0x1234)　y：4660(0x1234)　z：43981(0xabcd)
4)　x：10 y：2 z：3
5)　a[0]：'T' a[1]：'h' a[2]：'e' a[3]：' ' a[4] 以降：不定
　　b[0]：'C' b[1]：'o' b[2]：'m'b[3]：'p'
　　b[4]：'u' b[5]：'t' b[6]：'e' b[7]：'r'
　　b[8]：0 b[9] 以降：不定
6)　a[0]：'T' a[1]：'h' a[2]：0 a[3] 以降：不定
　　b[0]：'e' b[1]：0 b[2] 以降：不定
7)　a[0]：'a' a[1]：' ' a[2]：'b' a[3]：' a[4]：'b'
　　a[5]：' ' a[6]：'c' a[7]：0 a[8] 以降：不定
8)　a[0]：'n' a[1]：'e' a[2]：'w'a[3]：' '
　　a[4]：'w' a[5]：'o' a[6]：'r' a[7]：'l'
　　a[8]：0 a[9] 以降：不定

解説

1)　%＊dは456にマッチするが＊の指定により読み飛ばされて代入は行われない。変数yには次の789が代入される。ここで書式が尽きるので，変数zには何も代入されず前の値を保持する。

2)　標準入力の中で空白文字はデータの区切りになるが，空白文字がないからといってデータが区切られてないと考えることはできない。%2dなどのように変換仕様に入力幅が指定されたとき，その条件にマッチするデータが読み込めてしまえば，scanf関数は次の変換仕様の解釈に移ってしまう。いまは2桁ずつの読み込みであるから，標準入力の最初の12345は,

　12 34 5

というように分解されて読み込まれる。次の5678というデータとの間は空白で区切られているので，12345の最後の5と5678の最初の5が結合された55が変数zに代入されることはない。標準入力の中の空白文字には区切り文字としての意味があるが，書式文字列の中の空白には意味がない（無視される）ことにも注意。

3)　16進数の読み込み時にはデータの頭に0xまたは0Xが付いていてもいなくてもよい。入力されるデータを16進数とみなして解釈する。ただし，XCのVer.1では0xや0Xが付いた数字を認識できない。0の次のxやXを読んだ時点で変換が終了し，変数には0が代入される。

4)　書式文字列の中の空白文字は無視される。

5)　%cで変換するときには標準入力の空白文字は区切り文字としての意味を持たずそのまま配列に代入される。%sで変換するときには空白文字は区切り文字になるので，対応するComputerという文字列が入力幅の10文字に満たなく（8文字）ても，それだけが配列に代入される。そもそも，入力幅は読み込む文字の最大幅を指定するものであった。%cでは配列の最後にヌル文字（¥0）が代入されないが，%sでは配列の最後にヌル文字が代入されることにも注意。

6)　標準入力の空白文字は区切り文字になり，入力幅は読み込む最大文字数を示していることに注意。

7)　%[…]の機能は%sのサブセットと考えることができる。[ ]内に指定された文字以外を区切り文字として%sと同様の変換を行う。このとき空白文字は（[ ]に空白文字があれば）もはや区切り文字ではない。

8)　%[^…]は%[…]のまったく逆。[ ]内に指定された文字を区切り文字として%sと同様の変換を行う。

設問 2

1)5)9)11)12)が同じ（+000000123）
3)4)7)8)が同じ　（　　+123）

解説

2)6)10)はXCでは正しい変換仕様と認識されない。0を左詰めする指定は出力幅の指定にくっつけて%010dのように書く。.（ピリオド）の右で指定する出力する文字数はピリオドだけでは指定がなかったものと認識される。変換後の文字数が出力する文字数の指定より少ない場合はその数に達するまで左端から（小数点以下を示す場合は右端から）0で埋められる。

設問 3

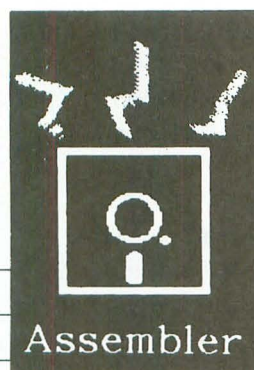
＊を出力幅の指定に書かず，ピリオドの右の出力文字数に＊を指定すればよい。たとえば,

printf(”%.＊d¥n”,5,10)；

などのように指定する。

グラフィックパターンの拡大・縮小

Murata Toshiyuki 村田 敏幸

前号の予告どおり，今回はグラフィックパターンの拡大・縮小です。基本的には，画質よりも描画速度を重視した拡大・縮小プットルーチンの作成ですが，おまけとして，速度よりも画質を優先した矩形領域の画像縮小ルーチンの例も用意しました。

今月はグラフィックパターンの拡大・縮小を取り上げる。ひとくちに拡大・縮小といっても目的に応じてプログラムの作り方は変わってくるものだ。ゲームに使うのであれば少々画質が落ちても実行速度が速いほうがいいし，グラフィックエディタに組み込むのなら速度よりも変換後の画質を重視したい。今回は質より実行速度を優先した拡大・縮小プットルーチンの作成を中心に話を進めていくことにするが，最後におまけとして，速度よりも画質を優先した矩形領域の画像縮小ルーチンの例も示すことにしよう。

座標の計算

グラフィックパターンの拡大・縮小は要は単純なスケーリング，比の問題だ。矩形領域(x0,y0)－(x1,y1)を領域(x2,y2)－(x3,y3)に写像する場合，元領域内のある点(x,y)は，

$$x'=\frac{x3-x2}{x1-x0}(x-x0)+x2$$

$$y'=\frac{y3-y2}{y1-y0}(y-y0)+y2$$

で求められる点(x',y')に対応する。ここで，

$$\frac{x3-x2}{x1-x0}=\text{横方向の拡大・縮小率}$$

$$\frac{y3-y2}{y1-y0}=\text{縦方向の拡大・縮小率}$$

だ。各点につき，上の式で対応する写像先の点を求めれば，任意の比率での拡大・縮小が行えるはずだ。試しにこのとおりX-BASICの関数にしてみるとリスト1のようになった。

リスト1は縮小時にはうまく動いているように見える。しかし，拡大時には描画結果が“すかすか”になるという症状を示す。考えてみれば当たり前のことで，リスト1では拡大元の矩形の大きさでループしているために，拡大先の領域を埋めるだけの点の数がないのだ。

また，一見問題なく見えた縮小時には描く点が多すぎ，重複して点を打っていることがわかる。本当は拡大・縮小先の矩形の大きさでループを組み，“拡大・縮小元の各点がどこに写るか”ではなく“拡大・縮小先の各点がどこから写ってくるか”を求めるのが正しい。拡大・縮小先の点(x',y')から逆に(x,y)を求め，(x,y)の色を拾って(x',y')に打つわけだ。見かけ上，拡大時は縮小の計算を，縮小時には拡大の計算をすることになる。

逆変換の式は先に出てきた式をちょっとひっくり返せば，

$$x=\frac{x1-x0}{x3-x2}(x'-x2)+x0$$

$$y=\frac{y1-y0}{y3-y2}(y'-y2)+y0$$

と得られ，プログラムのほうもリスト2となる。より正確な結果を得るためには，40行や60行の割り算の商を四捨五入すべきだが，その点さえ除けばリスト2は期待どおりの動作をする。

リスト1

```
 10 func test(x0,y0,x1,y1,x2,y2,x3,y3)
 20      int x,y,xx,yy,c
 30      for y=y0 to y1
 40           yy=(y-y0)*(y3-y2)/(y1-y0)+y2
 50           for x=x0 to x1
 60                xx=(x-x0)*(x3-x2)/(x1-x0)+x2
 70                c=point(x,y)
 80                pset(xx,yy,c)
 90           next
100      next
110 endfunc
```

リスト2

```
 10 func test(x0,y0,x1,y1,x2,y2,x3,y3)
 20      int x,y,xx,yy,c
 30      for yy=y2 to y3
 40           y=(yy-y2)*(y1-y0)/(y3-y2)+y0
 50           for xx=x2 to x3
 60                x=(xx-x2)*(x1-x0)/(x3-x2)+x0
 70                c=point(x,y)
 80                pset(xx,yy,c)
 90           next
100      next
110 endfunc
```

線分描画の応用

さて，マシン語にするうえでは，例によって実行速度を上げるために乗除算を使わずに済ませたい。実は，その方法を僕たちはすでに知っている。拡大・縮小の座標計算にはBresenhamの線分発生アルゴリズムがそっくりそのまま使えるのだ。つぎの式を見てもらいたい。

$$y=\frac{y1-y0}{x1-x0}(x-x0)+y0$$

これは2点(x0,y0)，(x1,y1)を通る直線の方程式だが，先に示した拡大・縮小時の座標計算の式とまったく同じ形をしている。そうとわかれば話は簡単，座標計算部分はBresenhamのアルゴリズムでx'からxを求めるループを組み，同様にy'からyを求めるループで括れば出来上がる。Bresenhamのアルゴリズムの2重ループだ。

リスト3にBresenhamのアルゴリズムを応用したマシン語版の拡大・縮小ルーチンを示そう。画面上の矩形領域を拡大・縮小すると領域の重なりに応じて処理を振り分ける必要が出てきて面倒なので，メインメモリ上に用意したグラフィックパターンを拡大・縮小して画面に描くプットルーチン仕様にした。なお，いまのところクリッピングはしていない。

リスト3のサブルーチンgxputには11～17行のような形式で用意した引数列の先頭アドレスをスタックに積んで渡す。また，グラフィックパターンは以前単純なプットルーチンを作ったときと同じ1ピクセル/1ワードでパレットコードを並べた形式を採用した。あのときも触れたように，この形式は扱いが楽な半面65536色モード以外で使うにはメモリ効率が悪いという問題がある。

今回はやらないが，256色モードや16色モードでは1ピクセル/1バイトや2ピクセル/1バイトにパックした形式（X-BASICのget，putの形式）のグラフィックパターンを受け取って，展開しつつ描画するようにしたほうがよい。

では，プログラムを順に見ていこう。

22行からサブルーチン本体が始まる。33行までで引数をレジスタに取り出す。描画先の矩形の座標をd0～d3，a1に描くグラフィックパターンの先頭アドレス，a2，a3にパターンの大きさだ。データレジスタが足りないものだから，a2，a3もデータ格納用に駆り出されている。そういえば，日頃はフレームポインタに使うa6もすでに動員されている。

36～39行で描画範囲の幅をd2，高さをd4に求めているが，この途中の37行と39行はこのプログラム最大の手抜きだ。あとの計算の都合上，描画範囲は最低縦横2ピクセルなくてはならないことになっており，その場逃れに1ピクセルしかないケースをここで弾いている。

43～47行でx座標について，Bresenhamのアルゴリズムを適用するのに必要なパラメータを計算する。誤差項の初期値をd0，誤差項の増分をa2，補正値をd2，ループカウンタの初期値をd4に求めている。

49～53行では同様にy座標についてのパラメータを計算し，あとは55行でグラフィックパターンの横1ライン分のバイト数をa4に求めた時点で描画の前処理が終わる。パターンの横1ライン分のサイズは“パターン横幅のピクセル数”の2倍（1ピクセル/1ワード）であり，すでにa2に“パターン横幅のピクセル数－1”の2倍が求まっているから，それ

リスト3　GXPUT.S（第1版）

```
 1: *         グラフィックパターンを拡大/縮小してプットする
 2:
 3:           .include        gconst.h
 4:           .include        gmacro.h
 5: *
 6:           .xdef   gxput
 7:           .xref   gramadr
 8: *
 9:           .offset 0
10: *
11: X0:       .ds.w   1           *描画先座標
12: Y0:       .ds.w   1           *
13: X1:       .ds.w   1           *
14: Y1:       .ds.w   1           *
15: PAT:      .ds.l   1           *パターンアドレス
16: XL:       .ds.w   1           *パターン横長さ-1
17: YL:       .ds.w   1           *パターン縦長さ-1
18: *
19:           .text
20:           .even
21: *
22: gxput:
23: ARGPTR = 4+8*4+7*4
24:           movem.l d0-d7/a0-a6,-(sp)
25:
26:           move.l  ARGPTR(sp),a6   *a6=引数列
27:
28:           movem.w (a6)+,d0-d3     *d0~d3=描画範囲の座標
29:           MINMAX  d0,d2           *x0≦x1を保証する
30:           MINMAX  d1,d3           *y0≦y1を保証する
31:
32:           movea.l (a6)+,a1        *a1=パターン先頭アドレス
33:           movem.w (a6)+,a2-a3     *a2=パターン横ピクセル数-1
34:                                   *a3=パターン縦ピクセル数-1
35:
36:           sub.w   d0,d2           *d2=描画範囲の横ピクセル数-1
37:           beq     done
38:           sub.w   d1,d3           *d3=描画範囲の縦ピクセル数-1
39:           beq     done
40:
41:           jsr     gramadr         *a0=描画先左上G-RAMアドレス
42:
43:           move.w  d2,d0           *
44:           neg.w   d0              *d0=xについての誤差項初期値
45:           move.w  d2,d4           *d4=xループカウンタ初期値
46:           add.w   d2,d2           *d2=xについての誤差項補正値
47:           add.w   a2,a2           *a2=xについての誤差項増分
48:
49:           move.w  d3,d1           *
50:           neg.w   d1              *d1=yについての誤差項初期値
51:           move.w  d3,d5           *d5=yループカウンタ初期値
52:           add.w   d3,d3           *d3=yについての誤差項補正値
53:           add.w   a3,a3           *a3=yについての誤差項増分
54:
55:           lea.l   2(a2),a4        *a4=パターン1ライン分バイト数
56:
57: yloop:    movea.l a0,a5           *a5=描画先
58:           movea.l a1,a6           *a6=参照元
59:
60:           move.w  d0,d6           *d6=x誤差項
61:           move.w  d4,d7           *d7=xループカウンタ
62: xloop:    move.w  (a6),(a5)+      *プロット
63:           add.w   a2,d6           *xについての誤差を累積させる
64:           bmi     xnext           *
65: xinclp:   addq.l  #2,a6           *参照元x座標を進める
66:           sub.w   d2,d6           *x誤差項を補正する
67:           bpl     xinclp          *誤差項が負になるまで繰り返す
68: xnext:    dbra    d7,xloop        *横幅分繰り返す
69:
70:           add.w   a3,d1           *yについての誤差を累積させる
71:           bmi     ynext           *
72: yinclp:   adda.l  a4,a1           *参照元y座標を進める
73:           sub.w   d3,d1           *y誤差項を補正する
74:           bpl     yinclp          *誤差項が負になるまで繰り返す
75:
76: ynext:    lea.l   GNBYTE(a0),a0   *描画先y座標を進める
77:           dbra    d5,yloop        *高さ分繰り返す
78:
79: done:     movem.l (sp)+,d0-d7/a0-a6
80:           rts
81:
82:           .end
```

に2を足してa4に代入している。

57行からがy座標についてのループだ。57～58行でつぎに描画する横1ラインの左端のアドレスをa5に，対応するパターン横1ライン分の左端アドレスをa6に入れておき，62～68行のx座標についてのループで1ライン分を描画する。62行でまず無条件に1ピクセル描く。a5はポストインクリメントされて2バイト進む。座標で考えると，これはx座標に1を足すことに相当する。それから63行で誤差項に増分を加える。その結果，誤差項が非負になったら“誤差が十分溜まった”ので，65～66行でパターンを指すポインタa6を1ピクセル分（2バイト）進め，同時に誤差項を補正する。

ここで，縮小時はポインタa6を数ピクセル分まとめて進める（途中を飛ばす）場合があることに注意しよう。数ピクセル分進めなければならないかどうかは，誤差項が1度の補正では負にならないことで判断がつく。そこで，65～67行のように誤差項が負になるまで，ポインタを進める処理と誤差項の補正をループで繰り返せばよい。

似たようなことは前回のソリッドスキャンコンバージョンでもやった。なお，このサブルーチンを拡大にしか使わないのであれば，数ピクセル飛ばす必要はないし，1度の補正で誤差項は負に戻るから67行の条件分岐命令は不要になる。

以上の処理を横幅分繰り返し，1ラインの描画が済んだら，70～74行で今度はy座標についてBresenhamのアルゴリズムによりパターンをつぎのラインに進めるかどうか，進めるとしたらどれだけ進めるかを判定する。やっていることは63～67行とほぼ同様だが，パターンを指すポインタは1ライン単位で進める（72行）。

yに関するループの最後の76行で次の描画に備えて描画先を1ライン下に進める。シンボルGNBYTEはgconst.hで定義された記号定数で，G-RAM横1ライン分のバイト数を表す。G-RAM上のアドレスにGNBYTEを足すことは，y座標に1を足すことに相当する。

ここまでの処理を描画範囲の高さ分繰り返して描画完了だ。

描画の高速化

一応動く版ができたところで質の向上を目指す。クリッピング処理の追加の前に無駄な処理を省いて軽く高速化してやろう。多重ループのプログラムでは，まず，最も内側のループに注目し，ループの外でできることをどんどん追い出すのが一番効く。

リスト3の場合，x座標に関してBresenhamのアルゴリズムを適用する62～68行のループだ。ループの中身についてはとくに無駄な処理をしている部分はない。ループ内で値の変わらない定数は前処理段階で計算してレジスタに入れてあるし，ループ構造もこれ以上簡潔にはならない。

が，このプログラムではここにBresenhamのアルゴリズムのループがあること自体が無駄なのだ。パターンへのポインタa6の進み方は毎回同じであり，このループを1回実行してみればどういうステップで進んでいくかがわかる。なにも，y座標が変わるごとに再計算する必要はない。

リスト4 GXPUT.S（若干の高速化版）

```
 1: *          グラフィックパターンを拡大/縮小してプットする
 2:
 3:            .include        gconst.h
 4:            .include        gmacro.h
 5: *
 6:            .xdef   gxput
 7:            .xref   gramadr
 8: *
 9:            .offset 0
10: *
11: X0:        .ds.w   1            *描画先座標
12: Y0:        .ds.w   1            *
13: X1:        .ds.w   1            *
14: Y1:        .ds.w   1            *
15: PAT:       .ds.l   1            *パターンアドレス
16: XL:        .ds.w   1            *パターン横長さ-1
17: YL:        .ds.w   1            *パターン縦長さ-1
18: TEMP:      .ds.l   1            *テーブル用
19: *
20:            .text
21:            .even
22: *
23: gxput:
24: ARGPTR = 4+8*4+7*4
25:            movem.l d0-d7/a0-a6,-(sp)
26:
27:            move.l  ARGPTR(sp),a6     *a6=引数受け渡し領域
28:
29:            movem.w (a6)+,d0-d3       *描画範囲の座標を取り出す
30:            MINMAX  d0,d2             *x0≦x1を保証する
31:            MINMAX  d1,d3             *y0≦y1を保証する
32:
33:            movea.l (a6)+,a1          *a1=パターン先頭アドレス
34:            movem.w (a6)+,a2-a3       *a2=パターン横ピクセル数-1
35:                                      *a3=パターン縦ピクセル数-1
36:            movea.l (a6),a6           *a6=テーブル用ワーク
37:
38:            sub.w   d0,d2             *d2=描画範囲の横ピクセル数-1
39:            beq     done
40:            sub.w   d1,d3             *d3=描画範囲の縦ピクセル数-1
41:            beq     done
42:
43:            jsr     gramadr           *a0=描画先左上G-RAMアドレス
44:
45:            move.w  d2,d0             *
46:            neg.w   d0                *d0=xについての誤差項初期値
47:            move.w  d2,d4             *d4=xループカウンタ初期値
48:            add.w   d2,d2             *d2=xについての誤差項増分
49:            add.w   a2,a2             *a2=xについての誤差項補正値
50:
51:            movea.l a6,a4             *a4=テーブル先頭アドレス
52:            move.w  d4,d5             *d5=xループカウンタ
53: iloop:     moveq.l #0,d1             *d1=xの増分(0に初期化)
54:            add.w   a2,d0             *xについての誤差を累積させる
55:            bmi     inext             *
56: iinclp:    addq.w  #2,d1             *xの増分を増す
57:            sub.w   d2,d0             *x誤差項を補正する
58:            bpl     iinclp            *誤差項が負になるまで繰り返す
59: inext:     move.w  d1,(a4)+          *テーブルに登録
60:            dbra    d5,iloop          *横幅分繰り返す
61:
62:            move.w  d3,d1             *
63:            neg.w   d1                *d1=yについての誤差項初期値
64:            move.w  d3,d5             *d5=yループカウンタ初期値
65:            add.w   d3,d3             *d3=yについての誤差項増分
66:            add.w   a3,a3             *a3=yについての誤差項補正値
67:
68:            addq.l  #2,a2             *a2=パターン1ライン分バイト数
69:
70:            move.l  a6,d2             *d2=テーブル先頭アドレス
71: yloop:     movea.l a0,a5             *a5=描画先
72:            movea.l a1,a6             *a6=参照元
73:
74:            movea.l d2,a4             *a4=テーブル先頭アドレス
75:            move.w  d4,d0             *d0=xループカウンタ
76: xloop:     move.w  (a6),(a5)+        *プロット
77:            adda.w  (a4)+,a6          *参照元x座標を進める
78:            dbra    d0,xloop          *横幅分繰り返す
79:
80:            add.w   a3,d1             *yについての誤差を累積させる
81:            bmi     ynext             *
82: yinclp:    adda.l  a2,a1             *参照元y座標を進める
83:            sub.w   d3,d1             *y誤差項を補正する
84:            bpl     yinclp            *誤差項が負になるまで繰り返す
85:
86: ynext:     lea.l   GNBYTE(a0),a0     *描画先y座標を進める
87:            dbra    d5,yloop          *高さ分繰り返す
88:
89: done:      movem.l (sp)+,d0-d7/a0-a6
90:            rts
91:
92:            .end
```

そこで，ループに入る前にあらかじめポインタの増分を求め，テーブルにしておくことを考える。たとえば，2倍に拡大する場合は同じ点を2度参照してとなりに進むので，ポインタの増分(バイト単位)は0，2，0，2……の繰り返しになる。また，2/3に縮小する場合であれば，パターンを2ピクセル拾ってから1ピクセル分飛ばすわけだから，ポインタの増分は2，4，2，2，4，2……の繰り返しだ。このような増分を並べたテーブルを先に作っておけば，横1ライン分を描くループが簡略化でき，実行速度もかなり違ってくるだろう。

その方針でリスト3を作り直すとリスト4のようになった。レジスタの使い方がちょこちょこと変わっていたりするので変更点のみではなく完全な形のリストで示す。適当に見比べてもらいたい。

テーブルを置くメモリはサブルーチンを呼び出す側が用意し，引数で渡すようにした。18行のように，引数列の末尾にこのワーク用メモリへのポインタが追加されている。ワークはスタック上にとってもよかったのだが，描画する範囲の横幅×2バイト，1024ドットモード時で最大2Kバイトの大きさになり，メインルーチン側ではそれを見越してスタックを確保しなければならない。それなら公に引数で渡すようにしても，メインの負担は同じようなものだ。

53～60行がテーブルを作成するループだ。リスト3の内側のループをそのまま抜き出したような形をしている。ただ，リスト3ではポインタを進めていた部分が増分をカウントするように変更されている。

飛んで76～78行がこのテーブルを作成して1ライン分を描くループだ。点を打ち，テーブルから引いた増分をポインタに足すだけに簡略化されたのがわかるだろう。

クリッピングの処理の追加

では，クリッピングの処理を追加しよう。プットルーチンのクリッピングでは描画範囲を切り詰めたら，同時にその分パターンのほうも切り捨てなければならない。ところが，いま作っているのは拡大・縮小つきのプットルーチンであり，どれだけパターンを切り捨てるかが簡単には求まらないという点でちょっとだけ難易度が高い。たとえば，クリッピングの結果30%描画範囲が狭まったら，パターンも30%切り捨てるわけだが，その30%が何ピクセルに相当するかはパターンの大きさや拡大・縮小率に応じて変わってくるのだ。

と書くと面倒そうだが，実をいうと似たような処理はすでにこの連載でも取り上げている。2つのパラメータからなり，片方をクリッピングしたらもう一方も同じ比率でクリッピングする処理，だ。拡大・縮小ルーチンとラインルーチンの対比からピンときたと思う。線分のクリッピングがまさにそうだ。x座標を切り詰めたら，同時にy座標も同じ比率で切り詰めていた。

で，リスト5がクリッピングの処理を追加した新版の拡大・縮小プットルーチンだ。変更点より追加部のほうが長いので，やはり完全なリストで掲載する。40～158行あたりがごっそり追加された部分だ。ここ以外では，8～9行に外部参照定義が追加され，185行（リスト4の68行に対応）に小さな修正が加わっている。

リスト5の40～48行では，完全にクリッピングウィンドウ外のケースを真っ先に弾いている。すると50行に到達した時点で描画範囲は（最低限，角は）クリッピングウィンドウの中にあるので，ここから本格的にクリッピングする。

51～55行で線分のクリッピングのときにもやったように座標に8000_Hのバイアスをかけて(ゲタを履かせて）大小関係を保ったまま無符号数に変換している。この変換は，平均を取るときのオーバーフローを防ぐのが目的だということを思い出そう。

続いて，パターンの大きさがアドレスレジスタに格納されたままでは演算がやりにくいので，57～58行でa2，a3を空いているデータレジスタd4，d5に転送する。

また，59，60行では描画範囲の右下隅の座標を保持するd2，d3をa4，a5に待避している。この時点で，データレジスタには次のような値が収まっている。

d0＝描画範囲の左上隅x座標（ゲタ履き）
d1＝描画範囲の左上隅y座標（ゲタ履き）
d2＝描画範囲の右下隅x座標（ゲタ履き）
d3＝描画範囲の右下隅y座標（ゲタ履き）
d4＝パターンの横幅－1
d5＝パターンの高さ－1

62～96行でウィンドウの左端で，描画範囲，グラフィックパターンともどもクリッピングする。これは，線分(d0,0)－(d2,d4)をウィンドウ左端のx座標minxでクリッピングする処理になぞらえることができる。ただし，d0，d2はゲタ履きなのにパターンの横幅d4にはゲタを履かせていないという点で少々変則的だ。グラフィックパターンは演算過程でオーバーフローするほど大きくはないだろうという悪い仮定をしている。その点さえ注意して見てもらえれば，以前の線分のクリッピングとまったく同じようなことをしているのがわかるだろう。なお，最初に作った版そのまま，描画先は最低2ピクセル幅が必要なので，変なタイミングではあるが，67～68行で1ピクセルにしかならないケースを弾いている。処理後，この疑似線分は(minx,?)－(d2,d4)のようにクリッピングされ，“？”の部分に“グラフィックパターンが何ピクセル分左側にはみ出すか”が求まる。

リスト5でいうと92行に達した時点でd3にこの値が収まっている。92行ではその分だけパターンの横幅を減じ，93～94行でパターンを実際に切り捨てている。98～134行のウィンドウ上端でのクリッピングもレジスタの割り当てが異なるだけでやっていることは同じだ。

ウィンドウの左端，上端でのクリッピングが済んだら制御は136行にくる。ラベルが示しているようにここでウィンドウ右端でクリッピングする。といっても，x座標についてのループカウンタのつじつまを合わせるだけだ。描画時の処理は左から右へと進むわけであり，処理開始位置である左端はきちんとクリッピングして正確な位置を求める必要があるが，処理終了位置である右端についてはループカウンタを適当に減らしてウィンドウの外まで描画しないよう調整すればクリッピングしたのと同じ効果がある。なまじクリッピングしない分，誤差の入り込む余地もない。144〜148行のウィンドウ下端でのクリッピングも同様だ。

ところが，残念ながらこの簡略クリッピングには副作用があることに気づいた。あとでBresenhamのアルゴリズムを適用する際にオーバーフローを生じる場合がある。具体的には描画範囲の右下隅のx座標，y座標のどちらか少なくとも一方が4000_Hよりも大きいと2倍する処理の段階で値が16ビットの負の数になり，アルゴリズムに混乱をきたす。バグも仕様書に書けば仕様に変わるというから作者である僕は放っておくが，読者はプログラムを修正することを考えてみてほしい。対処としてはクリッピングの処理を追加する正攻法と，Bresenhamのアルゴリズムを適用している部分をロングワードで演算するように修正する泥縄法の2通りがある。

さまざまな拡大・縮小ルーチン

ここから先はオプションだ。4本の拡大・縮小ルーチンのバリエーションを示す。

リスト6はリスト5の拡大時の処理をもう一段高速化した版だ(変更点のみ)。その代わり，縮小時には不要な処理を行うので，わずかに速度が低下する。高速化の理屈は単純で“横1ラインの内容が直前のラインと同一であれば真面目にループで描く代わりに1ラインコピーする”というものだ。

リスト7はリスト5を透明色に対応させた版だ(変更点のみ)。パターン中，パレットコードが0の部分は描画せずに下地をそのまま残す。ゲームのキャラクタ描画なんかに使えるかもしれない。もっとも，ゲームに使うのならより高速化すべきだろう。拡大・縮小率が決まりきった値しかとらないようなら，毎回の呼び出しごとに計算しているテーブルをあらかじめ用意して引数で渡すとか，拡大・縮小率に応じた専用ルーチンを用意するとか，ループを展開するとか，気の済むまでやればいい。もちろん，先に拡大・縮小したパターンを作成してメモリ上に持ってしまうのが一番速いが，その場合メモリ容量の問題もあって，あまり大きく拡大するのはあきらめなければならない。

メモリ容量といえば，最初にも触れたように，ゲーム向きの256色モードなどで使うのには今月のプログラムはメモリ効率が悪いから，その点もどうにかすることになるだろう。

一転してリスト8のg1to2は，質優先の画像1/2変換サブルーチンだ。ただし，65536色モード専用。絵を描く人ならこの種のルーチンは自前で用意していると思う。引数としては矩形範囲の座標（を格納したメモリへのポインタ）を与える。変換後の画像は元絵の左上隅の部分に描かれる。gxputのように点を間引いて縮小するのではなく，元絵の2×2の範囲をRGBごとに足して平均をとる。78〜84行あたりのコメントで殺してある行をすべて復活すると，平均をとった場合小数点以下を四捨五入するようになる。

リスト9のgshrinkはやはり質優先の画像縮小ルーチン（参考）で，こっちは縮小率を縦横個別に自由に指定できる。リスト8同様，縮小後の絵は元絵の左上隅に置かれる。半分以上冗談で作った工夫のかけらもないプログラムだが“注意して使えば”実用になるかもしれない。引数は30〜36行のような構造で用意して，その先頭アドレスをスタックに積んで渡す。TEMPは作業用に使うメモリで，最大6Kバイト必要だ。

このプログラムでは任意の縮小率を実現するために，各ピクセルを縮小率の分子に応じて等分して細かなサブピクセルに分割したうえで，得られたサブピクセルを縦横分母の数だけ集めて平均をとっている。内部でいったん元画像を縮小率の分子倍し，それから分母で割ると思えばいい。これを力ずくの4重ループで処理している。

このプログラムにおける縮小率とは，縮小先の画像の大きさと縮小元の画像の大きさの比ということになるが，安直に縮小先の大きさを分子，縮小元の大きさを分母とみなして約分せずにそのまま処理すると，とんでもなく遅くなるのは想像がつくだろう。

たとえば，512×512の範囲を256×256に縮小するときに，各ピクセルを縦横256等分し，1ピクセル描くごとにサブピクセルを512×512集めて平均をとっていたのでは日が暮れるどころの騒ぎではない。というわけで，冗談プログラムとはいえリスト9でも約分だけはちゃんとやっている。本筋ではないので細かくは触れないが，約分するのに必要な分子と分母の最大公約数はユークリッドの互除法と呼ばれるアルゴリズムで求めている。

で，ここまでいえば，このプログラムの使用上の注意は明らかだろう。“約分して分母が十分小さくなるような縮小率を選ばないと死ぬほど遅い”のだ。512×512の範囲を511×512に縮小するのに13分かかった。この場合，y方向については約分により縮小率1になるが，x方向の縮小率は511/512のままで，1点ごとに512個のサブピクセルの平均をとっていることになる。間違っても512×512の範囲を511×511に縮小するようなことはしないように。もしどうしてもそんなことをしたいのであれば（誰もしないって)，誤差は出るがx方向とy方向を個別に511/512に縮小して回避しよう。

ちなみに，リスト9は少しの修正で拡大にも使える。85〜88行のチェックを外し，159行の空行を，

lea.l　$c00000,a0

に変更する。この変更により拡大後の画像の描画位置は画面の左上に固定される。あとは，拡大後の画像が拡大元の画像をへたに上書きしてしまわないよう，元データをなるべく画面の右隅においてサブルーチンを呼び出せばよい。整数倍するときはただのモザイクになるので面白くもなんともないが，3/2倍などの非整数の倍率を指定した場合は（モザイクっぽさは残るものの）ある程度，色を補間する。

では，最後になったが，gxputの動作試験用ルーチンをリスト10に示す。あらかじめ65536色の画像を画面にロードしてから実行すると，画面中央の128×128の範囲を切り出し，256×256ドットモードでランダムな縮小・拡大率で描き続ける。

ついでにリスト11にg1to2とgshrinkのテストプログラムを示しておこう。画像をロードしてから走らせると，まず，gshrinkで3/4に縮小し，それをさらにg1to2で1/2にする。

＊

さて，拡大・縮小を片づけたところで，次回はグラフィックパターンを回転させてみることになるだろう。

リスト5　GXPUT.S（クリッピングつき）

```
  1: *         グラフィックパターンを拡大/縮小してプットする
  2:
  3:           .include        gconst.h
  4:           .include        gmacro.h
  5: *
  6:           .xdef   gxput
  7:           .xref   gramadr
  8:           .xref   cliprect
  9:           .xref   ucliprect
 10: *
 11:           .offset 0
 12: *
 13: X0:       .ds.w   1           *描画先座標
 14: Y0:       .ds.w   1           *
 15: X1:       .ds.w   1           *
 16: Y1:       .ds.w   1           *
 17: PAT:      .ds.l   1           *パターンアドレス
 18: XL:       .ds.w   1           *パターン横長さ-1
 19: YL:       .ds.w   1           *パターン縦長さ-1
 20: TEMP:     .ds.l   1           *テーブル用
 21: *
 22:           .text
 23:           .even
 24: *
 25: gxput:
 26: ARGPTR = 4+8*4+7*4
 27:           movem.l d0-d7/a0-a6,-(sp)
 28:
 29:           move.l  ARGPTR(sp),a6     *a6=引数受け渡し領域
 30:
 31:           movem.w (a6)+,d0-d3       *描画範囲の座標を取り出す
 32:           MINMAX  d0,d2             *x0≦x1を保証する
 33:           MINMAX  d1,d3             *y0≦y1を保証する
 34:
 35:           movea.l (a6)+,a1          *a1=パターン先頭アドレス
 36:           movem.w (a6)+,a2-a3       *a2=パターン横ピクセル数-1
 37:                                     *a3=パターン縦ピクセル数-1
 38:           movea.l (a6),a6           *a6=テーブル用ワーク
 39:
 40:           lea.l   cliprect,a0       *a0=クリッピング範囲
 41:           cmp.w   (a0)+,d2          *x1<minxならウィンドウ外
 42:           blt     done
 43:           cmp.w   (a0)+,d3          *y1<minyならウィンドウ外
 44:           blt     done
 45:           cmp.w   (a0)+,d0          *x0>maxxならウィンドウ外
 46:           bgt     done
 47:           cmp.w   (a0)+,d1          *y0>maxyならウィンドウ外
 48:           bgt     done
 49:
 50:           lea.l   ucliprect,a0      *a0=クリッピング範囲（ゲタ履き）
 51:           move.w  #$8000,d5         *x0,y0,x1,y1に8000Hのゲタを履かせる
 52:           add.w   d5,d0             *
 53:           add.w   d5,d1             *
 54:           add.w   d5,d2             *
 55:           add.w   d5,d3             *
 56:
 57:           move.w  a2,d4             *d4=パターン横ピクセル数-1
 58:           move.w  a3,d5             *d5=パターン横ピクセル数-1
 59:           move.w  d2,a4             *d2をa4に待避
 60:           move.w  d3,a5             *d3をa5に待避
 61:
 62: minxclip:                   *MINXでクリッピングする
 63:           move.w  (a0)+,d6          *d6=minx
 64:           cmp.w   d6,d0             *x0>minxなら
 65:           bcc     minyclip          *  クリッピング不要
 66:
 67:           cmp.w   d6,d2             *x1=minxなら
 68:           beq     done              *  描画幅が1ピクセルしかない
 69:
 70:           moveq.l #0,d3
 71: minxlp:   move.w  d0,d7
 72:           add.w   d2,d7
 73:           roxr.w  #1,d7
 74:           cmp.w   d6,d7
 75:           beq     xclipq
 76:           bcs     minx0
 77:
 78:           move.w  d7,d2
 79:           add.w   d3,d4
 80:           lsr.w   #1,d4
 81:           bra     minxlp
 82:
 83: minx0:    move.w  d7,d0
 84:           add.w   d4,d3
 85:           lsr.w   #1,d3
 86:           bra     minxlp
 87:
 88: xclipq:   move.w  d7,d0             *d0=x0=minx
 89:           add.w   d4,d3
 90:           lsr.w   #1,d3             *d3=パターンが左端にはみ出る分
 91:           move.w  a2,d4             *d4=パターン横ピクセル数-1
 92:           sub.w   d3,a2             *はみ出た分パターンの横幅を詰める
 93:           add.w   d3,d3             *
 94:           adda.w  d3,a1             *はみ出た分パターンを切り捨てる
 95:           move.w  a4,d2             *d2を復帰
 96:           move.w  a5,d3             *d3を復帰
 97:
 98: minyclip:                   *MINYでクリッピングする
 99:           addq.w  #1,d4             *
100:           add.w   d4,d4             *d4=パターン1ライン分のバイト数
101:
102:           move.w  (a0)+,d6
103:           cmp.w   d6,d1
104:           bcc     maxxclip
105:
106:           cmp.w   d6,d3
107:           beq     done
108:
109:           moveq.l #0,d2
110: minylp:   move.w  d1,d7
111:           add.w   d3,d7
112:           roxr.w  #1,d7
113:           cmp.w   d6,d7
114:           beq     yclipq
115:           bcs     miny0
116:
117:           move.w  d7,d3
118:           add.w   d2,d5
119:           lsr.w   #1,d5
120:           bra     minylp
121:
122: miny0:    move.w  d7,d1
123:           add.w   d5,d2
124:           lsr.w   #1,d2
125:           bra     minylp
126:
127: yclipq:   move.w  d7,d1
128:           add.w   d5,d2
129:           lsr.w   #1,d2
130:           sub.w   d2,a3
131:           mulu.w  d4,d2
132:           adda.l  d2,a1
133:           move.w  a4,d2
134:           move.w  a5,d3
135:
136: maxxclip:                   *MAXXでクリッピングする
137:           move.w  d4,d6             *d6=パターン1ライン分バイト数
138:
139:           move.w  (a0)+,d4          *d4=maxx
140:           sub.w   d2,d4             *d4=maxx-x1
141:           bmi     maxyclip          *
142:           moveq.l #0,d4             *右端にははみ出ていなかった
143:
144: maxyclip:
145:           move.w  (a0)+,d5          *d5=maxy
146:           sub.w   d3,d5             *d5=maxy-y1
147:           bmi     clipped
148:           moveq.l #0,d5             *下端にははみ出ていなかった
149:
150: clipped:
151:           sub.w   d0,d2             *d2=描画範囲の横ピクセル数-1
152:           beq     done
153:           sub.w   d1,d3             *d3=描画範囲の縦ピクセル数-1
154:           beq     done
155:
156:           move.w  #$8000,d7         *ゲタを脱がせる
157:           sub.w   d7,d0             *
158:           sub.w   d7,d1             *
159:
160:           jsr     gramadr           *a0=描画先左上G-RAMアドレス
161:
162:           move.w  d2,d0             *
163:           neg.w   d0                *d0=xについての誤差項初期値
164:           add.w   d2,d4             *d4=xループカウンタ初期値
165:           add.w   d2,d2             *d2=xについての誤差項増分
166:           add.w   a2,a2             *a2=xについての誤差項補正値
167:
168:           movea.l a6,a4             *a4=テーブル先頭アドレス
169:           move.w  d4,d7             *d7=xループカウンタ
170: iloop:    moveq.l #0,d1             *d1=xの増分(0に初期化)
```

```
171:         add.w   a2,d0            *xについての誤差を累積させる
172:         bmi     inext            *
173: iinclp: addq.w  #2,d1            *xの増分を増す
174:         sub.w   d2,d0            *x誤差項を補正する
175:         bpl     iinclp           *誤差項が負になるまで繰り返す
176: inext:  move.w  d1,(a4)+         *テーブルに登録
177:         dbra    d7,iloop         *横幅分繰り返す
178:
179:         move.w  d3,d1            *
180:         neg.w   d1               *d1=yについての誤差項初期値
181:         add.w   d3,d5            *d5=yループカウンタ初期値
182:         add.w   d3,d3            *d3=yについての誤差項増分
183:         add.w   a3,a3            *a3=yについての誤差項補正値
184:
185:         movea.w d6,a2            *a2=パターン1ライン分バイト数
186:
187:         move.l  a6,d2            *d2=テーブル先頭アドレス
188: yloop:  movea.l a0,a5            *a5=描画先
189:         movea.l a1,a6            *a6=参照元
190:
191:         movea.l d2,a4            *a4=テーブル先頭アドレス
192:         move.w  d4,d0            *d0=xループカウンタ
193: xloop:  move.w  (a6),(a5)+       *プロット
194:         adda.w  (a4)+,a6         *参照元x座標を進める
195:         dbra    d0,xloop         *横幅分繰り返す
196:
197:         add.w   a3,d1            *yについての誤差を累積させる
198:         bmi     ynext            *
199: yinclp: adda.l  a2,a1            *参照元y座標を進める
200:         sub.w   d3,d1            *y誤差項を補正する
201:         bpl     yinclp           *誤差項が負になるまで繰り返す
202:
203: ynext:  lea.l   GNBYTE(a0),a0    *描画先y座標を進める
204:         dbra    d5,yloop         *高さ分繰り返す
205:
206: done:   movem.l (sp)+,d0-d7/a0-a6
207:         rts
208:
209:         .end
```

リスト6 GXPUT.S（拡大時高速化版）

```
183:         add.w   a3,a3            *a3=yについての誤差項補正値
184:         move.w  a3,d7            *d7=yについての誤差項補正値
185:
186:         move.w  d4,d0
187:         addq.w  #1,d0
188:         lea.l   cnext(pc),a2     *a2=戻りアドレス
189:         bclr.l  #0,d0            *横ドット数は奇数か?
190:         beq     skip             *
191:         subq.l  #2,a2
192: skip:   lea.l   alincopy(pc),a3
193:         suba.w  d0,a3
194:
195:         move.l  a6,d2            *d2=テーブル先頭アドレス
196: yloop:  movea.l a0,a5            *a5=描画先
197:         movea.l a1,a6            *a6=参照元
198:
199:         movea.l d2,a4            *a4=テーブル先頭アドレス
200:         move.w  d4,d0            *d0=xループカウンタ
201: xloop:  move.w  (a6),(a5)+       *プロット
202:         adda.w  (a4)+,a6         *参照元x座標を進める
203:         dbra    d0,xloop         *横幅分繰り返す
204:
205:         add.w   d7,d1            *yについての誤差を累積させる
206:         bmi     lincpy
207: yinclp: adda.w  d6,a1            *参照元y座標を進める
208:         sub.w   d3,d1            *y誤差項を補正する
209:         bpl     yinclp           *誤差項が負になるまで繰り返す
210:
211: ynext:  lea.l   GNBYTE(a0),a0    *描画先y座標を進める
212:         dbra    d5,yloop         *高さ分繰り返す
213:
214: done:   movem.l (sp)+,d0-d7/a0-a6
215:         rts
216:
217: cpylp1: movea.l a0,a6
218:         lea.l   GNBYTE(a0),a0
219:         movea.l a0,a5
220:         jmp     (a3)
221:         move.w  (a6)+,(a5)+
222: cnext:  add.w   d7,d1
223:         bpl     yinclp
224: lincpy: dbra    d5,cpylp1
225:         bra     done
226:
227:         .dcb.w  GNPIXEL/2,$2ade  *move.l (a6)+,(a5)+
228: alincopy:
229:         jmp     (a2)
230:
231:         .end
```

リスト7 GXPUTON.S

```
  6:         .xdef   gxputon
 25: gxputon:
193: xloop:  move.w  (a6),d6          *d6=描画色
194:         beq     xskip            *透明ならプロットしない
195:         move.w  d6,(a5)+         *プロット
196:         adda.w  (a4)+,a6         *参照元y座標を進める
197:         dbra    d0,xloop         *横幅分繰り返す
198:         bra     xdone
199:
200: xskip:  addq.l  #2,a5            *描画先x座標を進める
201:         adda.w  (a4)+,a6         *参照元x座標を進める
202:         dbra    d0,xloop         *横幅分繰り返す
203:
204: xdone:  add.w   a3,d1            *yについての誤差を累積させる
205:         bmi     ynext            *
206: yinclp: adda.l  a2,a1            *参照元y座標を進める
207:         sub.w   d3,d1            *y誤差項を補正する
208:         bpl     yinclp           *誤差項が負になるまで繰り返す
209:
210: ynext:  lea.l   GNBYTE(a0),a0    *描画先y座標を進める
211:         dbra    d5,yloop         *高さ分繰り返す
212:
213: done:   movem.l (sp)+,d0-d7/a0-a6
214:         rts
215:
216:         .end
```

リスト8 G1TO2.S

```
 1: *       矩形領域を縦横1/2に縮小する
 2:
 3:         .include        gconst.h
 4:         .include        gmacro.h
 5: *
 6:         .xdef   g1to2
 7:         .xref   gramadr
 8:         .xref   gfclip
 9: *
10:         .offset 0
11: *
12: X0:     .ds.w   1
13: Y0:     .ds.w   1
14: X1:     .ds.w   1
15: Y1:     .ds.w   1
16: *
17:         .text
18:         .even
19: *
20: g1to2:
21: ARGPTR = 8
22:         link    a6,#0
23:         movem.l d0-d7/a0-a3,-(sp)
24:
25:         move.l  ARGPTR(a6),a1    *a1=引数列
26:         movem.w (a1),d0-d3       *d0～d3=座標
27:
28:         jsr     gfclip           *クリッピングする
29:         bne     done             *Z=0なら描画の必要なし
30:
31:         jsr     gramadr          *G-RAM上のアドレスを得る
32:
33:         sub.w   d0,d2            *d2=横ピクセル数-1
34:         sub.w   d1,d3            *d3=縦ピクセル数-1
35:
36:         addq.w  #1,d2            *横ピクセル数を半減
37:         lsr.w   #1,d2            *
38:         subq.w  #1,d2            *
39:         bmi     done             *
40:
41:         addq.w  #1,d3            *縦ピクセル数を半減
42:         lsr.w   #1,d3            *
43:         subq.w  #1,d3            *
44:         bmi     done             *
45:
46:         move.w  #GNBYTE,d1       *d1=ライン間のアドレスの差
47:
48:         movea.l a0,a2
49: loop1:  movea.l a0,a1            *a1=参照元ライン左端
50:         movea.l a2,a3            *a3=描画先ライン左端
51:         move.w  d2,d4            *d4=横ピクセル数-1
52:
53:         swap.w  d1
54:         swap.w  d2
55:         swap.w  d3
56:
57: loop2:  move.w  (a1)+,d0         *(x,y)の色を
58:         DERGB   d0,d1,d2,d3      * rgbに分解
59:
60:         move.w  (a1)+,d0         *(x+1,y)の色を
61:         DERGB   d0,d5,d6,d7      * rgbに分解して
62:         add.w   d5,d1            * 加算
63:         add.w   d6,d2            *
64:         add.w   d7,d3            *
65:
66:         move.w  GNBYTE-4(a1),d0  *(x,y+1)の色を
67:         DERGB   d0,d5,d6,d7      * rgbに分解して
68:         add.w   d5,d1            * 加算
69:         add.w   d6,d2            *
70:         add.w   d7,d3            *
71:
72:         move.w  GNBYTE-2(a1),d0  *(x+1,y+1)の色を
73:         DERGB   d0,d5,d6,d7      * rgbに分解して
74:         add.w   d5,d1            * 加算
75:         add.w   d6,d2            *
76:         add.w   d7,d3            *
77:
78: *       moveq.l #0,d0
79:         lsr.w   #2,d1            *b/4
80: *       addx.w  d0,d1
81:         lsr.w   #2,d2            *r/4
82: *       addx.w  d0,d2
```

```
 83:          lsr.w    #2,d3              *g/4
 84: *        addx.w   d0,d3
 85:
 86:          RGB      d1,d2,d3,d0        *カラーコードに再構成して
 87:          move.w   d0,(a3)+           *  プロット
 88:
 89:          dbra     d4,loop2           *横幅分繰り返す
 90:
 91:          swap.w   d1
 92:          swap.w   d2
 93:          swap.w   d3
 94:
 95:          adda.w   d1,a0              *参照元は2ライン下へ
 96:          adda.w   d1,a0              *
 97:          adda.w   d1,a2              *描画先は1ライン下へ
 98:
 99:          dbra     d3,loop1           *高さ分繰り返す
100:
101: done:    movem.l  (sp)+,d0-d7/a0-a3
102:          unlk     a6
103:          rts
104:
105:          .end
```

リスト9 GSHRINK.S

```
  1: *        グラフィック画像を縮小する
  2:
  3:          .include          gconst.h
  4:          .include          gmacro.h
  5: *
  6:          .xdef    gshrink
  7:          .xref    gramadr
  8:          .xref    gfclip
  9: *
 10: FPACK    macro    callno
 11:          .dc.w    callno
 12:          endm
 13:
 14: __UDIV   equ      $fe05
 15: *
 16: GCM      macro    M,N          *最大公約数を求めるマクロ
 17:          local    loop
 18:          move.w   N,d0
 19: loop:    moveq.l  #0,N
 20:          move.w   d0,N
 21:          divu.w   M,N
 22:          move.w   M,d0
 23:          swap.w   N
 24:          move.w   N,M
 25:          bne      loop
 26:          endm
 27: *
 28:          .offset 0
 29: *
 30: X0:      .ds.w    1            *描画先座標
 31: Y0:      .ds.w    1            *
 32: X1:      .ds.w    1            *
 33: Y1:      .ds.w    1            *
 34: XL:      .ds.w    1            *縮小後の横ピクセル数-1
 35: YL:      .ds.w    1            *縮小後の縦ピクセル数-1
 36: TEMP:    .ds.l    1            *作業用ワーク（最大6Kバイト）
 37: *
 38:          .offset -42
 39: *
 40: WORK:
 41: EX:      .ds.l    1
 42: DEX:     .ds.l    1
 43: CEX:     .ds.l    1
 44: EY:      .ds.l    1
 45: DEY:     .ds.l    1
 46: CEY:     .ds.l    1
 47: XSTEP:   .ds.w    1
 48: YSTEP:   .ds.w    1
 49: DENOM:   .ds.l    1
 50: DXL:     .ds.w    1
 51: SXL:     .ds.w    1
 52: PATSIZ:  .ds.w    1
 53: BUFF:    .ds.l    1
 54: _a6:     .ds.l    1       *±0
 55: _a7:     .ds.l    1
 56: ARGPTR:  .ds.l    1
 57: *
 58:          .text
 59:          .even
 60: *
 61: gshrink:
 62:          link     a6,#WORK
 63:          movem.l  d0-d7/a0-a5,-(sp)
 64:
 65:          move.l   ARGPTR(a6),a1     *a1=引数列
 66:
 67:          movem.w  (a1)+,d0-d3/a2-a3
 68:                                     *d0～d3=描画範囲の座標
 69:                                     *a2=縮小後横ピクセル数-1
 70:                                     *a3=縮小後縦ピクセル数-1
 71:          movea.l  (a1),a4           *a4=1ラインバッファ
 72:          move.l   a4,BUFF(a6)       *
 73:
 74:          jsr      gfclip            *クリッピングする
 75:          bne      done              *Z=0なら描画の必要なし
 76:
 77:          jsr      gramadr           *a0=描画先左上G-RAMアドレス
 78:          movea.l  a0,a1
 79:
 80:          sub.w    d0,d2             *d2=描画範囲の横ピクセル数-1
 81:          beq      done
 82:          sub.w    d1,d3             *d3=描画範囲の縦ピクセル数-1
 83:          beq      done
 84:
 85:          cmp.w    a2,d2             *拡大はできない
 86:          bcs      done              *
 87:          cmp.w    a3,d3             *
 88:          bcs      done              *
 89:
 90:          move.w   a2,d7             *
 91:          addq.w   #1,d7             *
 92:          add.w    d7,d7             *
 93:          move.w   d7,PATSIZ(a6)     *パターン1ラインのバイト数
 94:
 95:          move.w   d2,SXL(a6)        *縮小元横幅-1
 96:          move.w   a2,DXL(a6)        *xループカウンタ初期値
 97:          move.w   a3,d7             *d7=yループカウンタ初期値
 98:
 99:          swap.w   d7                *d7の下位ワードを待避
100:          move.w   d2,d6
101:          move.w   d3,d7
102:          addq.w   #1,a2             *a2=パターン
103:          addq.w   #1,a3
104:
105:          addq.w   #1,d2             *x方向縮小率を約分する
106:          move.w   d2,d1             *
107:          move.w   a2,d4             *
108:          GCM      d1,d4             *
109:          divu.w   d0,d2             *分子
110:          move.l   a2,d4             *
111:          divu.w   d0,d4             *分母
112:
113:          addq.w   #1,d3             *y方向縮小率を約分する
114:          move.w   d3,d1             *
115:          move.w   a3,d5             *
116:          GCM      d1,d5             *
117:          divu.w   d0,d3             *分子
118:          move.l   a3,d5             *
119:          divu.w   d0,d5             *分母
120:
121:          move.w   d2,d0             *
122:          mulu.w   d3,d0             *
123:          move.l   d0,DENOM(a6)      *平均するサブピクセル数
124:
125:          move.w   d2,d4
126:          subq.w   #1,d2             *平均するサブピクセルの
127:          move.w   d2,XSTEP(a6)      *  横方向の数-1
128:          move.w   d3,d5
129:          subq.w   #1,d3             *平均するサブピクセルの
130:          move.w   d3,YSTEP(a6)      *  縦方向の数-1
131:
132:          move.w   a2,d0             *xに関するパラメータ計算
133:          mulu.w   d4,d0             *←疑似拡大
134:          neg.l    d0                *
135:          move.l   d0,EX(a6)         *
136:          neg.l    d0                *
137:          add.l    d0,d0             *
138:          move.l   d0,CEX(a6)        *
139:          move.w   d6,d2             *
140:          add.l    d2,d2             *
141:          move.l   d2,DEX(a6)        *
142:
143:          move.w   a3,d0             *yに関するパラメータ計算
144:          mulu.w   d5,d0             *←疑似拡大
145:          neg.l    d0                *
146:          move.l   d0,EY(a6)         *
147:          neg.l    d0                *
148:          add.l    d0,d0             *
149:          move.l   d0,CEY(a6)        *
150:          move.w   d7,d3             *
151:          add.l    d3,d3             *
152:          move.l   d2,DEY(a6)        *
153:
154:          bsr      dergb             *最初のラインをrgbに分解
155:          movea.l  a5,a2             *a2=rgbごとの累算用バッファ
156:
157:          move.l   EY(a6),d5
158:          swap.w   d7                *d7を復帰
159:
160: yloop:   move.l   a0,-(sp)
161:
162:          moveq.l  #0,d0             *rgbごとの累算用バッファを
163:          movea.l  a2,a5             *  0で初期化
164:          move.w   DXL(a6),d4        *
165: clrlp:   move.l   d0,(a5)+          *
166:          move.l   d0,(a5)+          *
167:          move.l   d0,(a5)+          *
168:          dbra     d4,clrlp          *
169:
170:          move.w   YSTEP(a6),d6
171: yloop2:  movea.l  a2,a4
172:          movea.l  BUFF(a6),a5
173:
174:          swap.w   d6
175:          swap.w   d7
176:          movem.l  d5/a1-a3,-(sp)
177:
178:          movem.w  (a5)+,a0-a2       *a0～a2=参照する点のbrg
179:
180:          move.l   DEX(a6),d3
181:          move.l   CEX(a6),a3
182:          move.w   XSTEP(a6),d5
183:
184:          move.l   EX(a6),d4
185:          move.w   DXL(a6),d7
186: xloop:   moveq.l  #0,d0             *B
187:          moveq.l  #0,d1             *R
188:          moveq.l  #0,d2             *G
189:          move.w   d5,d6
190: xloop2:  add.l    a0,d0             *B
191:          add.l    a1,d1             *R
192:          add.l    a2,d2             *G
```

▶ Oh!Xを買い始めたときは，ただ「見る」という感じでパラパラめくってました。最近になって西川善司氏の文章を読んでみて「文章というのは宝島のようだ」と思うようになりました。「読む」ということがこんなに面白いとは感激です。　谷　聡雄(18)茨城県

```
193:
194:            add.l   d3,d4               *EX += DEX
195:            bmi     xnext2
196:            sub.l   a3,d4               *EX -= CEX
197:            movem.w (a5)+,a0-a2         *参照元x座標を進める
198: xnext2:    dbra    d6,xloop2
199:
200: xnext:     add.l   d0,(a4)+            *R
201:            add.l   d1,(a4)+            *G
202:            add.l   d2,(a4)+            *B
203:            dbra    d7,xloop
204:
205:            movem.l (sp)+,d5/a1-a3
206:            swap.w  d7
207:            swap.w  d6
208:
209:            add.l   DEY(a6),d5          *EY += DEY
210:            bmi     ynext2
211:            sub.l   CEY(a6),d5          *EY -= CEY
212:            lea.l   GNBYTE(a1),a1       *参照元y座標を進める
213:
214:            move.w  d6,d0               *処理完了寸前であれば
215:            or.w    d7,d0               *  この下のラインを
216:            beq     ynext               *  rgbに分解する必要はない
217:
218:            bsr     dergb               *1ライン分rgbに分解する
219:
220: ynext2:    dbra    d6,yloop2
221:
222: ynext:     movea.l (sp)+,a0
223:
224:            movea.l a0,a3
225:            movea.l a2,a5
226:            move.l  DENOM(a6),d1        *d1=平均の分母
227:
228:            move.w  DXL(a6),d6
229: drawlp:    move.l  (a5)+,d0
230:            FPACK   __UDIV
231:            move.w  d0,d2               *B
232:            move.l  (a5)+,d0
233:            FPACK   __UDIV
234:            move.w  d0,d3               *R
235:            move.l  (a5)+,d0
236:            FPACK   __UDIV              *G
237:            move.w  d0,d4
238:            RGB     d2,d3,d4,d0         *カラーコードに構成して
239:            move.w  d0,(a3)+            *  プロット
240:            dbra    d6,drawlp           *横幅分繰り返す
241:
242:            lea.l   GNBYTE(a0),a0       *描画先y座標を進める
243:            dbra    d7,yloop            *高さ分繰り返す
244:
245: done:      movem.l (sp)+,d0-d6/a0-a5
246:            unlk    a6
247:            rts
248:
249: *
250: *          1ライン分rgbに分解して
251: *          バッファに展開する
252: dergb:
253:            movea.l a1,a3
254:            movea.l BUFF(a6),a5
255:            move.w  SXL(a6),d4
256: dergb0:    move.w  (a3)+,d0
257:            DERGB   d0,d1,d2,d3
258:            movem.w d1-d3,(a5)
259:            addq.l  #6,a5
260:            dbra    d4,dergb0
261:            rts
262:
263:            .end
```

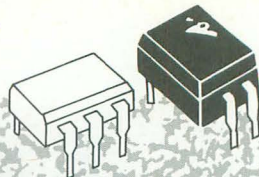
リスト10 TEST. S

```
 1: *       gxputのテスト用プログラム
 2: *
 3:         .include        doscall.mac
 4:         .include        iocscall.mac
 5: *
 6:         .xref   gxput
 7: *       .xref   gxputon
 8: *       .xref   gget
 9:         .xref   setcliprect
10: *
11: FPACK   macro   callno
12:         .dc.w   callno
13:         endm
14: *
15: __RAND          equ     $fe0e
16: __LTOS          EQU     $fe11
17: *
18:         .text
19:         .even
20: *
21: ent:
22:         lea.l   inisp,sp
23:
24:         clr.l   -(sp)
25:         DOS     _SUPER
26:
27:         lea.l   argbf2,a1
28:         IOCS    _GETGRM
29:
30:         pea     window
31:         jsr     setcliprect
32:         addq.l  #4,sp
33:
34:         moveq.l #14,d1  *256x256,65536
35:         IOCS    _CRTMOD
36:         IOCS    _G_CLR_ON
37:
38:         lea.l   argbuf,a1
39:         pea.l   (a1)
40:
41: loop:   lea.l   argbuf,a0
42:
43:         moveq.l #4-1,d1
44: loop2:  FPACK   __RAND
45:         lsr.w   #7,d0
46:         move.w  d0,(a0)+
47:         dbra    d1,loop2
48:
49:         jsr     gxput
50:
51:         DOS     _KEYSNS
52:         tst.w   d0
53:         beq     loop
54:         DOS     _INKEY
55:         cmpi.b  #$20,d0
56:         bne     done
57:
58: pause:  DOS     _INKEY
59:         cmpi.b  #$20,d0
60:         beq     loop
61: done:   move.l  #$0010_0000,-(sp)
62:         DOS     _CONCTRL
63:         DOS     _EXIT
64: *
65:         .data
66:         .even
67: *
68: argbuf: .dc.w   0,0,511,511
69:         .dc.l   pat
70:         .dc.w   128-1,128-1
71:         .dc.l   temp
72: *
73: argbf2: .dc.w   192,192,319,319
74:         .dc.l   pat
75:         .dc.l   pate
76: *
77: window: .dc.w   16,16,255-16,255-16
78: *
79:         .bss
80:         .even
81: *
82: pat:    .ds.w   128*128
83: pate:
84: temp:   .ds.w   512
85: *
86:         .stack
87:         .even
88: *
89:         .ds.l   4096
90: inisp:
91:         .end    ent
```

リスト11 TEST2. S

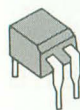
```
 1: *       glto2, gshrinkのテスト用
 2: *
 3:         .include        doscall.mac
 4: *
 5:         .xref   glto2
 6:         .xref   gshrink
 7: *
 8:         .text
 9:         .even
10: *
11: ent:
12:         lea.l   inisp,sp
13:         clr.l   -(sp)
14:         DOS     _SUPER
15:
16:         pea.l   argbuf
17:         jsr     gshrink
18:         addq.l  #4,sp
19:
20:         pea.l   argbf2
21:         jsr     glto2
22:         addq.l  #4,sp
23:
24:         DOS     _EXIT
25: *
26: argbuf:                 *gshrinkの引数
27:         .dc.w   0,0,511,511     *3/4
28:         .dc.w   383,383         *
29:         .dc.l   temp
30: *
31: argbf2:                 *glto2の引数
32:         .dc.w   0,0,511,511
33: *
34:         .bss
35:         .even
36: *
37: temp:   .ds.w   1024*3
38: *
39:         .stack
40:         .even
41: *
42:         .ds.l   4096
43: inisp:
44:         .end    ent
```

▶ 視力が落ちたおかげで目にする物体すべてをアンチエリアシングした状態で見られる。なんか得した気分。　井田　奨一(23)東京都

ハードウェア工作入門《14》

メカトロニクス制御(その4)

Misawa Kazuhiko
三沢 和彦

今月はステッピングモーターをキャタピラタンクに組み込んで実際に工作して動かすことを考えていきます。ところが三沢氏はちょっと不満顔。どうやら持ち前のプライドが妥協を許さず来月号からハイテクキャタピラタンク編を開始する予定です。

前回は簡単なステッピングモーターの制御を試みてみました。今月は実際に模型を工作して，その中にステッピングモーターを組み込む実験を行ってみようと思います。

ステッピングモーターと模型の組み合わせ

ステッピングモーターを模型の中に組み込む際に最初に考えることは，どのようにして模型の動力部分とモーターのシャフト（軸）とを連結させるかということです。私が，秋月電子で入手したステッピングモーター(芝浦S4H40B06F-01)には，シャフトの先に直径15mmの25枚ギアがすでについていました。電動モーターを使った自動車模型などでは，モーターの回転を車輪に伝達するギヤボックスというものがよく使われています。そこで，今回はステッピングモーターを自動車模型の駆動に使ってみようと思いました。

ところで，皆さんの手元にあるステッピングモーターにギヤがついていなかったらどうしましょうか。そういうときは，

1)　自分で工夫してギヤを取り付ける
2)　ギヤ付きのモーターに買い換える
3)　今回の工作は見送って，次回のメカトロニクス発展編を待つ

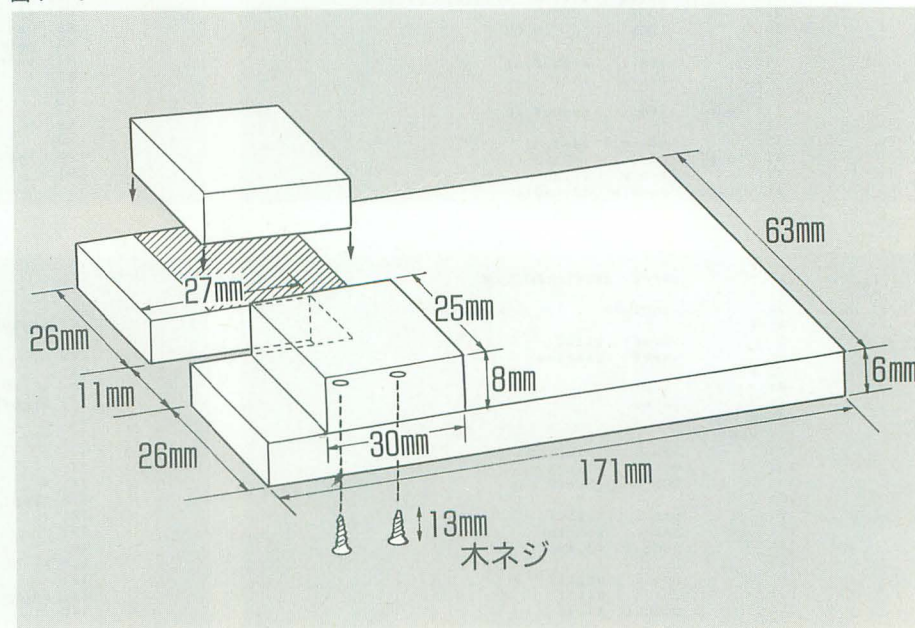
の3つが考えられます。

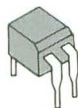
実は，今回の応用編で完成した模型に対して私自身が不満の点がありました。この不満点を解消するために来月号から，高速直流モーターを制御する目的でインタフェイス設計製作と実際の模型工作のための「ハードウェア工作入門ハイテクキャタピラタンク編」を開始する予定なのです。

話を戻して，モーター制御を模型工作に応用することを考えましょう。まず考えられるのは，一般に市販されているプラモデルを改造するというやり方です。ところが，プラモデルに組み込まれているモーターは今回使っているステッピングモーターよりもかなり小さく，せっかくのプラモデルでもシャーシに収まらないため不格好になってしまいます。幸い，模型用ギヤボックスはバラ売りの部品として手に入りますから，それを使って模型自体も自作してしまおうというわけです。

ハードウェア工作模型編

模型の世界では一番のメーカーとして，「田宮模型」があります。ギヤボックスも田宮模型製のものは用途に応じて種類も豊富にあり，一般の模型屋でも簡単にそろいます。私は今回の模型の部品をすべて「東急ハンズ」渋谷店（☎03-5489-5119）で購入しました。ギヤボックスだけでも，高速用，強力タイプ，3段変速，リモコン用などなど，どれを選んだらよいか迷ってしまうぐらいです。さらにギヤボックス単体だけではなく，自動車工作セット，F1カー工作セット，あるいはタンク工作セットとい

図1ーa

って，木製のシャーシとホイール，タイヤ，電池ケース，スイッチなど最低限の部品の入ったキットまであるのです。このキットはきわめて完成型に近いプラモデルと違って，部品の配置などはかなり自由にレイアウトできるようになっていますし，シャーシを改造するのにもスペース的に十分です。そこで今回はリモコン制御にぴったりなタンク工作基本セット（楽しい工作シリーズNo.29)を選びました。これはかなり遊べる模型だと思います。ではこれから，このタンク工作基本セットの組み立てとその改造方法を順番に説明していきましょう。

ただしここで注意が必要です。今回は改造を施すために少し余計な部品が必要なのです。必要なのは，厚さ8mm，縦30mm，横25mmの木片２切れとそれを本体シャーシ（これはタンク工作基本セットについてくる）に取り付けるための太さ2.4mm,長さ10mm程度の木ネジ４個です。これらの部品も東急ハンズで購入しました。実際には厚さ8mm，幅30mm，長さ450mmの木の板を買って，鋸で25mmの部品２個を切り出しました。部品がそろったら，いよいよ工作です。

模型タンクの製作

まずパッケージを開けて，取扱説明書から読み始めることにします。工作に必要な工具（ドライバー，錐，ペンチ，カッターナイフなど）をそろえ，部品に足りないものがないことを確認したら，いよいよ工作開始です。まず，木製のシャーシ板から加工を始めましょう。今回取り付ける予定のステッピングモーターのシャフトは高さが20mmなのに対して，付属のギヤボックスのギヤの高さは12mmなので，そのままでは高さが合いません。ステッピングモーターとギヤボックスとを取り付けるためには，8mm厚の木片をギヤボックスの下に敷いて高さを合わせる必要があるのです。

シャーシ板は図１のような寸法になっていて，用意した厚さ8mm，縦30mm，横25mmの木片を図の位置に取り付けます。取り付けには，裏面から太さ2.4mm，長さ10mm程度の木ネジで１個につき２カ所ずつ固定します。このとき注意しなければならないのは，木ねじをねじ込んでいく箇所にあらかじめ錐で穴を開けておくということです。太さ2mmもある木ねじを工作用木材に無理やり突っ込んでいくと板を割ってしまうからです。逆に錐の穴が大きすぎてもねじが固定されませんから，十分気をつけてください。そしてその上にギヤボックスを木ねじで固定します。ギヤボックスは，まずそれ自体を組み立てなければなりませんが説明書をよく見てやればなんの問題もないでしょう。そのあと，セットに付属の小さい木ねじで固定します。固定位置は寸法図（図1-a,b）を参考にしてください。

ギヤボックスが固定できたら，次はそれに合わせてステッピングモーターを固定します。ところが，前回の製作ではステッピングモーターのコードはインタフェイス基板に直接取り付けてしまっているので，そのままでは取り付けにも不便ですし，第一リモコンで遠隔操作することもできません。そこで，あとから延長ケーブルと接続コネクタを製作することにして，今はコード６本すべてを真ん中あたりからちょん切ってしまいます。ただし，あとでチェックが必要なので，７月号のmotor.basが確実に動作することを確認してから，コードを切り離してください。

さて，自由になったステッピングモータ

図１－ｂ　ギヤボックスの取り付け

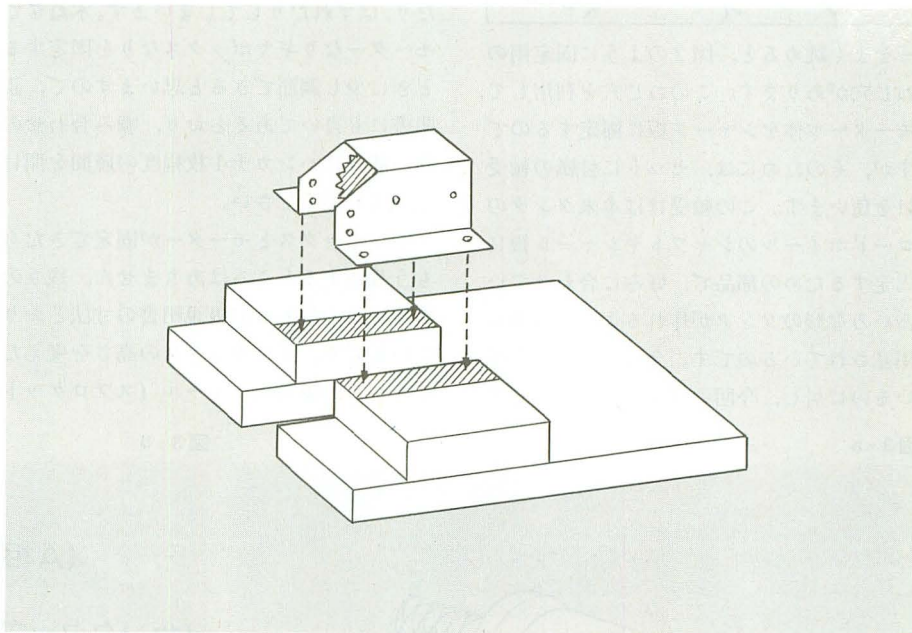
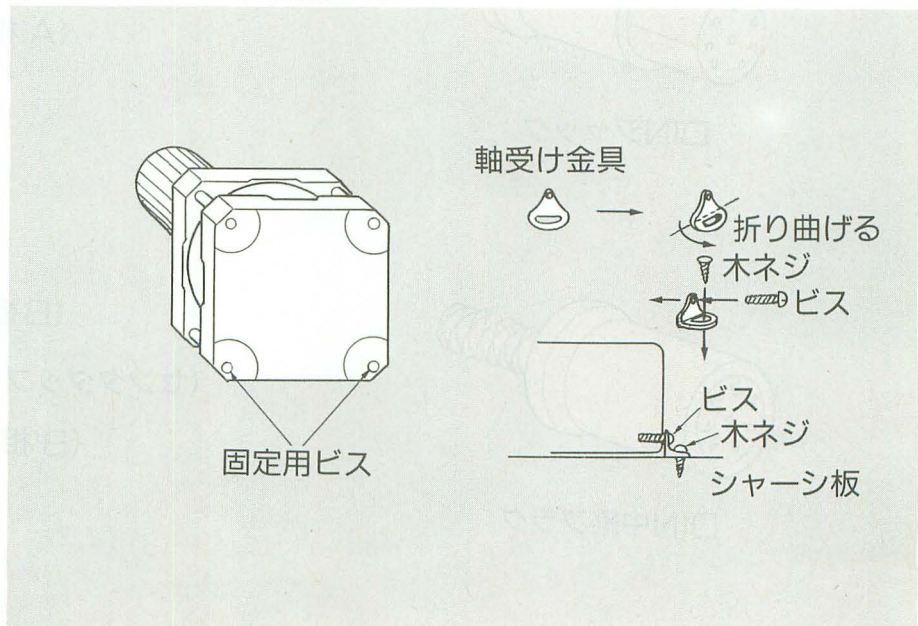
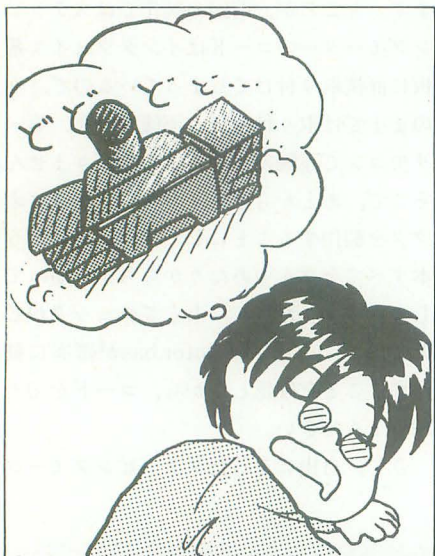
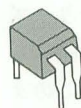



図２　ステッピングモーターの固定

ーをよく眺めると，図2のように固定用のねじ穴があります。このねじ穴を利用して，モーター本体をシャーシ板に固定するのですが，そのためには，セットに付属の軸受けを使います。この軸受けは本来タンクのロードホイールのシャフトをシャーシ板に固定するための部品で，好みに合わせていろいろな形のタンクが作れるように多めに用意されているのです。全部で10個入っているのに対し，今回のタンクでは6輪しか使わないので4個余ります。このうちの2個を図2のように取り付けます。手順としては，もともとモーターの外部ケースを固定していたビスを2本はずし，そのビスを軸受け部品の小さい円形の穴に通したあと，再びモーターのビス穴に固定します。ペンチで細長い穴の部分をつまんで，しっかり直角に折り曲げます。そして，付属の小さい木ねじで折り曲げた箇所の細長い穴を通してシャーシ板に固定するのです。

ここで特に注意すべきことは，ギヤの噛み合わせです。あまりきつく当ててしまうと抵抗が大きくなって滑らかに動きません。逆にゆるすぎても遊びが大きくて擦り減ったり，はずれたりしてしまいます。木ねじでモーターなりギヤボックスなりを固定するときに少し調節できると思いますので，説明書にも書いてあるとおり，噛み合わせの間の部分にハンカチ1枚程度の隙間を開けて固定してください。

ギヤボックスとモーターが固定できたら，もう改造するところはありません。残りのホイールの位置は取扱説明書の寸法どおりでOKです。ギヤボックスの高さを変えたことから，駆動用ホイール（スプロケットホイール）の高さも変わっていますが，キャタピラの穴にスプロケットホイールの突起をしっかりはめ込み，キャタピラをしっかり張ってからアイドラホイールのシャフト受けの位置を決めれば，まったく問題ありません。それから，スイッチ，電池ケースとそれらの間のコード配線は必要ないので省略します。

モーター用延長ケーブルの製作

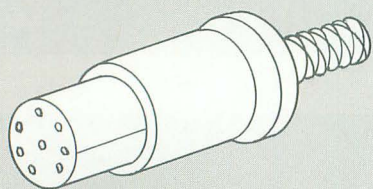
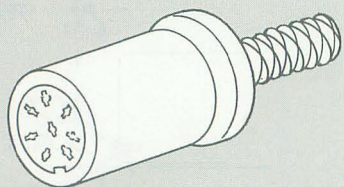
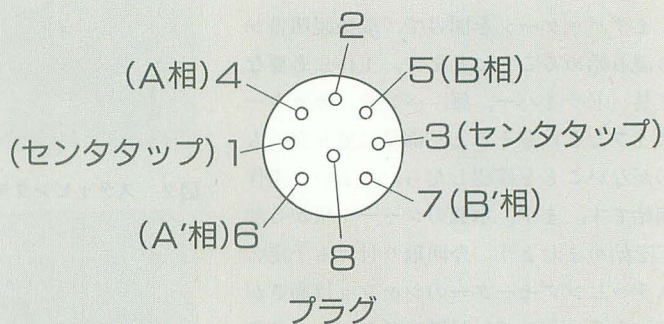
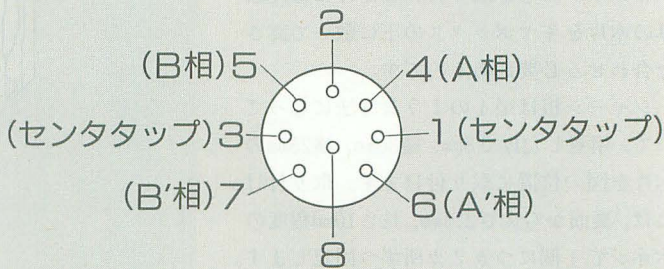
さて，モーターのコード6本を切り離してしまったので，再びインタフェイス基板に接続しなければなりません。切り離したコードの間は5mほどの延長ケーブルでつなぎますが，今回の延長ケーブルは10ピンフラットケーブルを使います。実際のモーターのコードは6本なのでフラットケーブルを縦に裂いて6ピンケーブルにしてもよいのですが，他の用途にも兼用するために私は2本を裂いて8ピンケーブルにしてあります。皆さんは10ピンのままにしておいてもかまいません。さて，この延長ケーブルをモーターのコードにどうつなぐか考えます。もちろん，6本ずつ計12カ所を直接

図3-a

DINジャック

DIN中継プラグ

図3-b

(A相)4
2
5(B相)
(センタタップ)1
3(センタタップ)
(A'相)6
8
7(B'相)
プラグ

(B相)5
2
4(A相)
(センタタップ)3
1(センタタップ)
(B'相)7
8
6(A'相)
ジャック

ハンダ付けする手もありますが，今回は別の用途も考えてコネクタで簡単に着脱できるように設計しました。コネクタは８ピンDINプラグ・中継ジャックといわれるものです（図3-a)。接続部分を見てピンが立っているほうがプラグ，穴の開いているほうがジャックです。いつものT-ZONEパーツショップにはこのDIN中継ジャックは置いてありませんでしたので，私は秋葉原のヒロセムセンパーツショップ(☎03-3255-2211)で８ピンDINプラグ・中継ジャックを手に入れました。もしこの部品が身近に手に入らなければ，ヒロセムセンパーツショップでは通信販売も行っているようですので，問い合わせてみてください。

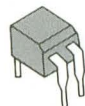
では，コネクタ工作について説明していきましょう。最初にモーター側のコードに中継ジャックをつなぎます。中継ジャックのカバーをはずすと黒いプラスチックの部品と金具の部品２個とに分かれます。このうち黒いプラスチックのジャック本体をハンダ付けする端子の側から見たのが図3-bです。端子番号が１～８までついていますが，今回はA相（４番），B相（５番），A'相（６番），B'相（７番）にハンダ付けします。また，センタータップからの共通端子の２本は１番と３番にハンダ付けします。今回は，２番と８番は使わないのでそのままにしておきます。これとまったく同じ配線を今度はインタフェイス基板から出ているコードについても行います。モーターのコードは色分けされているので，各相の独立端子および共通端子の対応を間違えないようにして，プラグの端子にハンダ付けしていきます。プラグのほうにも端子番号が付いているのでよく見てください。

注意しなければならないのは，プラグとジャックとで端子は左右対称になっているという点です。端子番号はきちんと左右対称に付いているので確認してください。それからもうひとつ，絶対に忘れてはならないのは，ハンダ付けの前にあらかじめカバーにコードを通しておくことです。すべてハンダ付けが終わってしまってから，カバーにコードを通すことはできません。

最後に延長ケーブルの部分です。フラットケーブルの両端に１組のプラグとジャックを取り付けるのですが，フラットケーブルの端から順番に端子番号にしたがってハンダ付けしていきます。このとき，両端のプラグとジャックとで端子番号が対応していないとトラブルを起こしますので気をつけてください。端子がすべてハンダ付けできたら，金具部品と組み合わせてカバーに収めます。カバーする前に，金属金具の一番端の部分をペンチでつまんでコードを束ねておくのがよいでしょう。

コネクタケーブルのハンダ付けが終わったら，それをつないでいよいよmotor.basをRUNさせてみましょう。コードを切り離す前とまったく同じ動作をするかを確認してください。もし，モーターの回転が滑らかでなかったり，まったく回転しなかったりしたときはコネクタへのハンダ付けの組み合わせに入れ違いなどのミスが考えられます。全部で６本しかないのですから，あわてずに１本１本たどっていきましょう。延長ケーブルの前と後で同じ色のコードにつながっていかなければなりません。

さて，正常にハンダ付けができていればモーターが滑らかに回りだし，ギヤを伝わってキャタピラが動きだすはずです。もしモーターはちゃんと回っているのにキャタピラが滑らかに動かないときはモーターとギヤボックスの噛み合わせに問題があります。motor.basでモーターを回転させながらギヤをよく観察してみて，動きが突っかかるようであれば，モーターを押しつけすぎで，逆に空回りしているようであれば，離しすぎです。モーターやギヤボックスをシャーシ板に固定している木ねじをゆるめて，調節し直してください。

インタフェイス部分，メカニクス部分ともに異常なければ，タンクは地上をゆっくり走るはずです。駆動はmotor.basで十分でしょう。好みにしたがってリストを書き換えてみてください。タンクは平地を走らせるだけではつまらないので，何か，本や板切れ，棒など障害物を置いてみましょう。たいていの障害物ならそれを力強く乗り越えていきます。私は，模型のなかでも特にキャタピラタンクがかっこいいと思います。確かに高速のF1マシンも魅力的ですが，それとはまたひと味違うおもしろさがキャタピラタンクにはあると思います。

次回への課題

ところが，私自身しばらく遊んでいるとすぐ飽きてしまいました。その理由としては，

1)　走行スピードが大変遅い。これは，ステッピングモーターの回転速度が一般の模型用モーターの回転速度より遅いために，それをさらにギヤで減速してしまうとキャタピラの動くスピードが大変遅くなってしまうからである

2)　曲がれない。モーター１個で左右両輪を駆動しているために，前後の動きしかできない

そこで，これらの欠点を克服しようと思い，最高性能のキャタピラタンクを設計することにしました。これまでどおりステッピングモーターを使ってもよいのですが，モーター自体が大きくて，ひとつのシャーシに２個載せるとなると全体がかなりの大きさになってしまいます。そこで，コンパクトにしかも高速で前後進だけでなく左右旋回までできるように，模型用の直流モーターを使ったタンクを製作実習していこうと思います。手順としては，まず直流モーターをパソコンでコントロールするインタフェイスを設計，製作します。そして，今回のタンクをさらに改造して新たな動力メカを組み込み，それを操縦するプログラムを完成させていきます。そのあとは……このタンクにマル秘システムを搭載してハイテクマシンに仕上げる計画です。何がハイテクマシンか皆さんで予想を立ててみてください。

では来月以降，理論編から実習編，応用編，発展編，完成編と全５回を予定していますので，ぜひ楽しみにしていてください。

表１　部品表

品名	数量	価格
タンク工作基本セット（楽しい工作シリーズNo.29）	１セット	700円
工作材木（厚さ８㎜，幅30㎜，長さ450㎜）	１本	90円
木ねじ（太さ2.4㎜，長さ13㎜）	１袋	130円
10Pフラットケーブル	５m	@100円
８ピンDINプラグ	２個	@135円
８ピンDIN中継ジャック	２個	@170円

★(で)のショートプロぱーてぃ その23

放物線も再帰も算数

Komura Satoshi 古村 聡

数学なんてキライだあ，という（で）氏の意思を無視するように今月は次のようなラインアップとなりました。ともにX68000用で，放物線を描いていく点を制御するゲームと，いろいろなマンデルブロ集合を表示してくれるデモです。

illustration : T.Takahashi

ああ，眠い。暑いし，眠いし。皆様，くそ暑い毎日ごきげんいかがでしょうか。私が暑苦しさで評判の(で)でございます。

それにしても暑いですね。私がこれを書いている梅雨の最中でも，この暑さだもんなあ。これから1カ月先なんて想像もつきませんや。

ところで，夏といえば夏休みの宿題。いやいや，私も苦しめられました。「夏休みの友」とかいうドリル。国語やら算数やらの練習問題がびっしりあるんですよね。

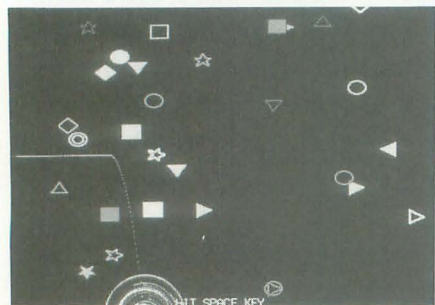
しかし，あれってかえって逆効果なんじゃないですかね。だって夏休みですよ。昼寝が夏休みの風物詩じゃないですか。算数のドリルを見る。暑い。だらける。ぐう。……ほら，昼寝に突入しちゃった。

自慢じゃないけど，私なんか義務教育9年間のおかげで，数学の本見ると眠るくせがついちゃったんですよ。こうでしょ。数学の本を開くと，……ぐう。

放物線はアップダウン

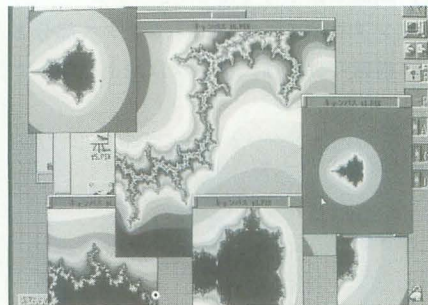
今月の1本目はX-BASIC用のアクションゲームです。

UPDOWN.BAS for X68000
(X-BASIC)
兵庫県 山田 智史

ルールは簡単。ドットを操作し，左端のスタートから右端のゴール目指してつっぱしれ。それだけです。操作キーもとても少ないんです。カーソルキーの上と下だけ，なのであります（スタートのときにスペースキーも押すけど）。

しかし，ルールがやさしくて操作キーも少ないからやさしい，とはいかないのであります。ふにゃ。結構むずかしいゲームだなあ。何がむずかしいって，あんた，カーソルキー押しても思ったようにいかんのですよ。このゲームはゴールに向かって，障害物を避けて進んでいく，本当にただそれだけ，それだけなんですよ。ああ，それなのに，それなのに。なにゆえ，この私の指はドットをゴールへと導いていけないのでありましょう。こ，この私に，この私になぜできないんだあ，“うっきーっ!!”，って猿化してどうする（最近は西川善司化するともいうらしいが）。

ふつうのゲームだと，キーを押すとそのとおりに動く。あと，ショートプロだと慣性が働いて，そのままずるずる上下に動いてしまうなんてのが結構多い。これはひねくれてる（と個人的には思う）。

しかしですね，このゲームの場合，上下のカーソルキーを押すと，ストーンっと放物線を描いて飛んで，あるいは落ちてしまうんですね。これが。いや，こいつの場合は落ちるってより，「墜ちる」のほうが似合ってるかな。むむむむむ……。とにかくやってみたまえ。むずかしいぞお。

ま，反射神経のない私のような人間にはちょっと難易度が高いというのはともかく，このゲームはとても完成度が高いと思います。プログラムリストも短く，読みやすい。そして，なんといっても音楽の効果的な使い方がいいです。ショートプロのゲームというと効果音がないのが多いんですが（私がハンズで作ったのもそうだった)，これは違います。ゲーム中の音楽のテンポを変えたものを，ゲームオーバーの音楽に使ったりしているのです。

それと，ん？ なになに，“もう少し付加機能をつけたかったのですが，長くなるのでやめました。ネームエントリーや，コンティニューはすぐできると思いますので，皆さんで作ってみてください。また，効果音をつければ面白さ倍増です。SOUND文があればいいんですが。できる方はやってみてください。Y座標に合わせて，音程を上下させるだけです”（投稿原稿より）。

そうですねえ。いちおう，m_vset()とかでFM音源の音色を変えることはできるけど，直接レジスタをいじるSOUND文みたいなのはないもんね。m_play()で音を出すのも，ちょっとずつ音色を変えるのは苦しいし。なんかいい方法はないもんでしょうかね。

また，アカデミック

続きまして，今月の2本目になりますのはこれも同じくX-BASIC用のグラフィックデモ（?）プログラムなのであります。

MAND.BAS for X68000
(X-BASIC,要SX-WINDOW)
東京都 小島 昇

前回，数式からグラフを描いてくれるプログラムがありましたが，今月は特定の数式である図形を描いてくれるプログラムなのです。2カ月連続でアカデミックなショートプロなんですね。で，それは何かともうしますと，グラフィック関係ではとても有名なマンデルブロ集合を描いてくれるプ

UPDOWN.BAS

MAND.BAS

ログラムなのです。

では，例によって使い方から。

このプログラムはBASICなのでそのまま打ち込んでRUNするだけでOK。ただし，セーブした図形をまた見るためにはSX-WINDOWが必要になります。

まず，プログラムを打ち込んでください。で，実行。RUNすると，絵のX方向の大きさ（何ドットか，だね），Y方向の大きさ，画面の左上のX座標，Y座標，それから1ドットの間隔(分解能ってやつね)を聞いてくるので，入れてやります（ここでリターンだけだとプログラム終了。パラメータを入れ間違えたときには分解能の入力のときにリターンだけ入れるようにすれば，もう一度はじめから設定しなおすことができます）。たとえば，Xが128，Yが64，始点が(0,0)，分解能が0.00000125とかね。

待つこと数時間から数日でその図形が出来上がる，てなわけです。

計算が終わるとセーブするファイル名を聞いてきますので，ファイル名を拡張子なしで入れてください。自動的にPIX形式でセーブされ，また最初に戻ります。リターンのみのときはセーブせず，また最初に戻ります。

セーブした絵を見るにはSX-WINDOWを使ってください。PIX形式になっているので，その絵のアイコンをクリックするだけで見ることができます。

コンパイラでかっとべ

このプログラムは実行すると終了までにかなり時間がかかるんですよね。BASICのままだと朝セットして，1日たって，次の日の夜には半分ちょっと見えてきたかなあ，という感じです。

もちろんプログラムの性質上，時間がかかるのはしかたないんですが，何日もX68000を占有されるというのもちょっと困りますよね。

ところがところが，Cコンパイラのパッケージを持ってる人なら，コンパイルすれば数十倍は速くなっちゃうんですよね。

Cを持っている皆さんは（というか，この際，持ってない人は買ってでも）コンパイルしてから実行してください。なんたって劇的に計算時間が違いますから。

コンパイルの方法はいたって簡単。Cコンパイラをインストールしてあるなら，BASICに入って，リストを打ち込み（一度RUNして，正しく動いているかを確認してからね），

```
save "mand"
```

としてセーブします。そのあと，

```
A>CC MAND.BAS
```

とするだけで，ディスク上にmand.xというファイルを作ってくれます。そこで，

```
A>MAND
```

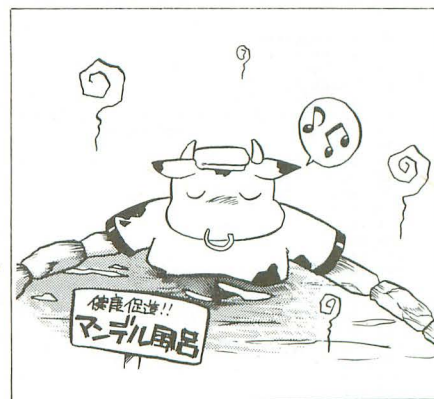
と打つと，BASICの数十倍速いスピードで実行してくれるってわけです。簡単でしょ？

GCCを使ってさらに高速なプログラムにすることもできます。ちなみに私はGCC1.39を使って，

```
A>bc mand.bas
A>gcc-O-fomit-frame-pointer-finline - functions - fstrength - reduce
mand.c baslib.a iocslib.a doslib.a
```

とコンパイルして，さらにIOCS.Xとデバイスドライバにfloat2.x ver.2.0を使って……，とかなり速くしてしまいました。ふふふふ。計ってはいないんだけど5割以上速いんじゃないかな？

これで計算に日単位かかってたものがひと晩寝れば図形ができているようになるわけです。もう，ほくほくってすよね。ふふふふ。

さあ，Cコンパイラを持ってない人は食事の回数を減らして金をためてでも買うんだ(すっかりシャープのまわしものモード)。

アカデミックは永遠に

実行してくれた方はわかる（あ，写真でもわかるか)でしょうけど，とてもきれいな図形ができるんですね。なんか前衛芸術だといってもばれなさそうな感じの図形です。これがある式によるグラフだなんて，信じられないですよね。

このマンデルブロ集合っていうのはなんでも幾何では有名な複素平面集合で，zとαっていう複素数の変数を使って，$z=\alpha$をまず初期値に使って，

$f(z)=z^2+\alpha$

という式にぶちこんで，その結果を画面に描いて，出てきたf(x)をまたzに入れてぶちこんで，と繰り返すとああいう図形になるんだそうなんですね。ううむ，数学の不思議。私もよくわからないんでほとんど小島さんの投稿原稿丸写しなんだけど。

でも，よくわからないじゃあまずいので，図形を描かせている間にもう一度そのテの本をよく読んで理解することにしよう。ぽちぽち，と数字をセットして，と。えーと，なになにf(z)がz^2で……，ぐう。

……はっ，もうとっくに図形ができている。やっぱりコンパイルすると速いなあ。おあとがよろしいようで(つくてんてんてん，と)。

リスト1　UPDOWN.BAS

```
 10 /*************************************************************
 20 /*  UP-DOWN for X-BASIC by S.Yamada (Mar. 29th, 1991) *
 30 /*************************************************************
 40 /*
 50 int round=1,hi_score=0,score=0,level=3
 60 /*
 70 initialize()
 80 while 1
 90  round_init()
100  if play_game()=0 then round=round+1
110 endwhile
120 end
130 /*
140 func play_game()
150  int x,miss=0
160  float y=256#,vy,vvy
170  str in
180  /*
190  m_tempo(180)
200  vpage(&B1000)
210  apage(3)
220  for x=8 to 511
230   in=inkey$(0)
240   if in=chr$(31) then vvy=level/10#
250   if in=chr$(30) then vvy=-level/10#
260   vy=vy+vvy
270   y=y+vy
280   score=score+abs(vy*10)
290   if score>hi_score then hi_score=score
300   if (point(x,y)<>0)or(y<0)or(y>511) then {
310    bang(x,y)
320    miss=1
330    score=0
340    round=1
350    break }
360   pset(x,y,15)
370  next
380  if miss=0 then success()
390  return(miss)
400 endfunc
410 /*
420 func bang(x,y)
430  int i
440  /*
```

```
450  m_tempo(64)
460  m_assign(1,9)
470  m_play()
480  vpage(&B1100)
490  for i=0 to 127
500   circle(x,y,i,rnd()*15+1,rnd()*361,rnd()*361,360)
510   if inkey$(0)=" " then break
520  next
530  print_score()
540  vpage(&B1101)
550  while inkey$(0)<>" ":endwhile
560  m_assign(1,1)
570  m_play()
580 endfunc
590 /*
600 func success()
610  m_tempo(200)
620  print_score()
630  vpage(&B1111)
640  while 1
650   palet(1,rnd()*65536)
660   if inkey$(0)=" " then break
670  endwhile
680 endfunc
690 /*
700 func round_init()
710  int i,x,y,c,d
720  str ch
730  /*
740  m_tempo(96)
750  print_score()
760  vpage(&B1001)
770  apage(3)
780  wipe()
790  line(0,0,511,0,15)
800  line(0,511,511,511,15)
810  for i=1 to round*10+20
820   x=rnd()*432+56
830   y=rnd()*462+25
840   ch=chr$(&H81)+chr$(&H99+rnd()*13)
850   c=(i mod 7+1)*2+1
860   d=rnd()*4
870   kput(x,y,ch,1,1,2,c,d)
880  next
890  line(0,256,7,256,5)
900  vpage(&B1101)
910  while inkey$(0)<>" ":endwhile
920 endfunc
930 /*
940 func initialize()
950  int i
960  /*
970  screen 1,1,1,1
980  console 0,32,0
990  locate ,,0
1000  /*
1010  i=atoi(right$(time$,2))
1020  i=i*atoi(mid$(time$,4,2))
1030  i=i*atoi(left$(time$,2))
1040  randomize((i mod 65536)-32768)
1050  /*
1060  set_bgm()
1070  m_tempo(96)
1080  m_play()
1090  vpage(&B0)
1100  apage(0)
1110  kput(212,160,"ROUND:",1,1,1,15,0)
1120  kput(188,244,"HI-SCORE:",1,1,1,15,0)
1130  kput(212,328,"SCORE:",1,1,1,15,0)
1140  apage(1)
1150  symbol(128,128,"CONGRATURATIONS!",2,1,1,1,0)
1160  symbol(129,128,"CONGRATURATIONS!",2,1,1,1,0)
1170  apage(2)
1180  kput(200,480,"HIT SPACE KEY",1,1,1,13,0)
1190 endfunc
1200 /*
1210 func print_score()
1220  apage(0)
1230  fill(262,0,511,511,0)
1240  kput(262,160,itoa(round),1,1,1,15,0)
1250  kput(262,244,itoa(hi_score),1,1,1,15,0)
1260  kput(262,328,itoa(score),1,1,1,15,0)
1270  apage(3)
1280 endfunc
1290 /*
1300 func kput(x,y,s;str,bx,by,sz,pal,dir)
1310  int i,j
1320  /*
1330  for i=0 to 2
1340   for j=0 to 2
1350    symbol(x+i,y+j,s,bx,by,sz,pal-1,dir)
1360   next
1370  next
1380  symbol(x+1,y+1,s,bx,by,sz,pal,dir)
1390 endfunc
1400 /*
1410 func set_bgm()
1420  int i
1430  /*
1440  m_init():for i=1 to 9:m_alloc(i,100):next
1450  for i=1 to 6:m_assign(i,i):next
1460  m_tempo(150)
1470  m_trk(1,"@8v1418o4q4")
1480  m_trk(2,"@7v1418o5q2p1")
1490  m_trk(3,"@57v1418o4q2p2")
1500  m_trk(4,"@58v1318o5q4")
1510  m_trk(5,"@9v1514o3q4")
1520  m_trk(6,"v15116o1")
1530  m_trk(9,"@8v1418o4q4")
1540  m_trk(1,"[do]c{cc}8ccececd{dd}8dd>ggab<[loop]")
1550  m_trk(2,"[do]rcrcrcrcrdrdrdrd[loop]")
1560  m_trk(3,"[do]rgrgrgrgrgrgrgrg[loop]")
1570  m_trk(4,"[do]{grgg}4[loop]")
1580  m_trk(5,"[do]c>g<c>b8<c8d>a<d>a8b8<[loop]")
1590  m_trk(6,"[do]@59cc@45c@59c[loop]")
1600  m_trk(9,"[do]c{cc}8cce-ce-cd{dd}8dd>gga-b<[loop]")
1610 endfunc
1620 /*
1630 /*  お疲れ様でした・・・
```

リスト2 MAND.BAS

```
10 /*===============================================
20 /*         mand.bas  Ver1.0                 1991/05/06
30 /*         マンデルブロ集合表示プログラム
40 /*===============================================
50 float za,zb,st,xorg,yorg
60 dim int col(15)
70 dim char heder(47)={'T','X','1','6',0,0,&H80,&H28,0,0,0,0,1,0,1,0}
80 str pna
90 int xmax,ymax,stat
100 screen 2,0,1,1 : console ,,0
110 /*         パレットの設定
120 col(0)=rgb(0,0,0)          /* 黒
130 col(1)=rgb(10,0,20)        /* マゼンタ・
140 col(2)=rgb(10,0,31)        /* 青
150 col(3)=rgb(15,15,31)       /* 水色
160 col(4)=rgb(24,24,31)       /* 水色
170 col(5)=rgb(31,31,31)       /* 白
180 col(6)=rgb(18,31,18)       /* 緑
190 col(7)=rgb(0,29,4)         /* 緑
200 col(8)=rgb(20,31,0)        /* 緑黄
210 col(9)=rgb(31,31,0)        /* 黄
220 col(10)=rgb(31,24,0)       /* 黄だいだい
230 col(11)=rgb(31,16,0)       /* だいだい
240 col(12)=rgb(28,10,0)       /* 赤だいだい
250 col(13)=rgb(28,0,0)        /* 赤
260 col(14)=rgb(20,0,0)        /* 茶
270 col(15)=rgb(10,0,0)        /* 焦げ茶
280   for n=0 to 15            /* ヘッダーに色コードをセット
290   palet(n,col(n))
300   heder(n*2+16)=col(n) shr 8
310   heder(n*2+17)=col(n) and 255
320   next
330   repeat
340     repeat
350     cls : wipe()          /* パラメータの入力
360     locate 0,24
370     input"X方向のビット数=",xmax
380     if xmax=0 then end
390     input"Y方向のビット数=",ymax
400     if ymax=0 then end
410     input"表示位置X     =",xorg
420     input"表示位置Y     =",yorg
430     input"分解能        =",st
440     until st>0
450   pna=" "
460     if dspmand(xmax,ymax,xorg,yorg,st)=0 then {  /* マンデルブロの表示
470     print chr$(&H1E)+chr$(5);
480       repeat
490       stat=0
500       input "ファイル名     =",pna
510       if pna<>"" then stat=pix(xmax,ymax,pna)    /* ファイルセーブ
520       if stat<>0 then print chr$(&H1E)+chr$(&H1E)+chr$(&H1A); : beep
530         until stat=0
540     } else{
550     beep
560     }
570   until 0
580 end
590 /*----------------------------------------
600 /* 画面のビットイメージをPIX形式でセーブする。
610 /* あらかじめheder()にパレット情報をセットしておくこと
620 /* 引き数 xs:X方向のドット数
630 /*        ys:Y方向のドット数
640 /*        na:ファイル名
650 /* 返り値 エラーコード
660 /*----------------------------------------
670 func pix(xs;int,ys;int,na;str)
680 int x,y,i,fp
690 char c=0,d=1
700 print "セーブ中です.."；
710 if xs<128 or xs>640 then print : return(-1) /* 画面の最小は128*64
720 if ys<64 or ys>480 then print : return(-1)  /* 画面の最大は640*480
730 /* ファイルサイズをヘッダーに代入する。
740 if (xs and &HF)<>0 then x=(xs and &HFFF0)+&H10 else x=xs
750 i=x*ys¥2+40         /* SX-WINDOWはヘッダーに書かれたファイルサイズと物理的な
760 heder(4)=i shr 24 /* ファイルサイズが異なってもエラーメッセージを出さない。
770 heder(5)=(i shr 16)and &HFF
780 heder(6)=(i shr 8)and &HFF
790 heder(7)=i and &HFF
800 /* 画面のサイズをヘッダーに代入する。
810 heder(12)=xs shr 8 : heder(13)=xs and &HFF
820 heder(14)=ys shr 8 : heder(15)=ys and &HFF : xs=x
830 /* ヘッダーをディスクに書き込む
840 fp=fopen(na+".PIX","c")
850 if fp<0 then print : return(-1)
860   for i=0 to 47
870   if fputc(heder(i),fp)<0 then {
880     fclose(fp) : print : return(-1)
890     }
900   next
910 /* ビットイメージをデイスクに書きこむ。
920   for i=0 to 3
930     for y=0 to ys-1
940       for x=0 to xs-1
950       if(point(x,y)and d)<>0 then c=c or 1
960         if(x and 7)=7 then {
970         if fputc(c,fp)<0 then fclose(fp) : print : return(-1)
980         c=0
990         } else {
1000        c=c shl 1
1010        }
1020      next
1030    print using"####%";(ys*i+y+1)*100/(ys*4); /* 処理率の表示
1040    print string$(5,chr$(&H1D));
```

```
1050      next
1060    d=d shl 1
1070    next
1080 fclose(fp) : print
1090 return(0)
1100 endfunc
1110 /*------------------------------------
1120 /*  マンデルブロ集合を表示する。
1130 /*  引き数  (xs, ys)表示するビット数
1140 /*            (xo, yo)左上の座標  s分解能
1150 /*  返り値  エラーコード
1160 /*------------------------------------
1170 func dspmand(xs;int,ys;int,xo;float,yo;float,s;float)
1180 int x,y,n,ov,nmax
1190 float   ca,cb,zabs
1200 nmax=((int(1/s) shl 4)and &HFFFFFFF0)+31
1210 print "計算中です...";
1220 if xs<128 or xs>640 then print : return(-1)  /* 引き数のチェック
1230 if ys<64 or ys>480 or s<=0 then print : return(-1)
1240    for y=0 to ys-1
1250    cb=yo-s*y
1260      for x=0 to xs-1
1270      ca=xo+s*x
1280      ov=0 : za=ca : zb=cb
1290        for n=1 to nmax
1300        mand(ca,cb)
1310          if za>5 or zb>5 or za<-5 or zb<-5 then {
1320          ov=n : n=nmax
1330          } else {
1340          zabs=za*za+zb*zb              /* 絶対値が5を越えたか調べる。
1350          if zabs>25 then ov=n : n=nmax
1360          }
1370        next
1380      pset(x,y,ov and 15) /* 絶対値を5をこえるまでの計算回数により色をセット
1390      next
1400    print using"####%";(y+1)*100¥ys; /*  処理率の表示
1410    print strings(5,chr$(&H1D));
1420    next
1430 print
1440 return(0)
1450 endfunc
1460 /*----------------------------------------
1470 /*  f(z)=Z^2+Aの計算をする。
1480 /*  引き数 z=za+jzb   a=a+jb
1490 /*  帰り値 z=za+jzb
1500 /*----------------------------------------
1510 func mand(a;float,b;float)
1520 float   r,x
1530 r=za*za-zb*zb+a
1540 x=2*za*zb+b
1550 za=r : zb=x
1560 endfunc
```

(で)のぱーてぃハンズ第3部——(その2)

遊び心はぺけろく

先月号では画面周りを作ったのであります。んで，肝心のゲームを進めるところなんですけど，たとえば，人生ゲームとかあのテのゲーム盤が目の前にあるとしますね。さて，あなたはどうしますか？

答え：ゲームをする

ううむ，なんかアドベンチャーゲームのように強引になってしまったが，そういうわけでこれからプレイヤーさんにプレイしてもらうのです。だんだんゲームっぽくなってきたぞ？ 当たり前のことだけど。

はてさて，プレイヤーさんにプレイしてもらうとしても，このゲームは2人ゲームなので，先攻，後攻を決めてもらわなくてはいけないんですよね。ということでそういう関数を作ります。

*

plyselect( ):
　プレイヤー1は人であると表示
　プレイヤー1の変数に1(=人)を代入
　プレイヤー2は人か，コンピュータか？
　プレイヤー2の変数にどちらかを入れる
　プレイヤー1，2どちらが先攻かを聞く
　変数cplayerにセット

*

はい，あがりっと。画面にメッセージを表示して，キー入力を待って変数に入れるだけなんだから簡単，簡単。

あ，今月のリストではプレイヤー2のときにも人しか入らないようになっています。また，作るのはプレイヤー1が人，2が人かコンピュータってことにしてあるけど，のちのちプレイヤー1，2どちらがコンピュータでもいいように作ってあります。

思考ルーチンを作りさえすれば，プレイヤー2用をプレイヤー1用に変更するのはそんなにむずかしくありませんからね。だから，思考ルーチンを作り終わったら，今度はプレイヤー1用の思考ルーチンを作ってみて，プレイヤー1の思考ルーチン対プレイヤー2の思考ルーチンなんてやってみるのも面白いんじゃないかな，なんて思います。人間はヒマになっちゃうけど（せっかく作った思考ルーチンに仲間はずれになるのは悲しいっ！ とか）。

プレイヤーセレクトはそんなところ(なにせリストがあれだもんな)。

教育的指導

さてさて，続いては人に手を打ってもらって，それを表示するのだ。このとき注意しなければならないことがあるんですね。それは，

“人はルールを守らない”

ってことなんですね。そうなんですよ，スピード違反はしょっちゅう，駐車違反，横断歩道は手をあげて渡らないし，歩くときはつい左側を歩いてしまう。あ，これは交通ルールの話か。

そういう意味ではなくて，ゲームをやっているときには結構人間はルール間違いを起こすんですよね。それではゲームにならない。

そこで“手を人に打ってもらうときにそれをやっていいかどうかチェックする”ということをしなくちゃいかんのですね。

このゲームのルールでは横1行か縦1列の中で，もう選択されたところ以外から選んでもらいます。つまり，選んだものは“もう打てないマス”に変化するんですよね。将棋なんかだとできるところが膨大にあるのでむずかしいけど，ここではわりと手がかぎられているので，「できないこと，できること」を画面で表示するのも簡単そうです。

ということで，このゲームの仕様は以下のように決めました。

1)　選べるマスは色が変わる
2)　選べるマスを表示することでチェックの代わりにする

これなら打ち間違いをすることも少なさそうです。

では，次にプレイヤー1の入力ルーチン，msell( )の説明。

*

msell( )
　まず，選べるマスのうちいちばん最初のマスを調べて，そこをカーソル位置に
　プレイヤー1の選べる行の色を変え，数値を書く
　カーソル位置をさらに色を変えて表示
　リターンキーが押されたなら決定
　そうでなければカーソルを次に移動

*

とこんなことをしています。カーソル入力ルーチンとルール判定ルーチンが一緒になっていますが，ちゃんとしたゲームを作るときなどは，ここはちゃんと分離させたほうがいいのではないかと思います。

そうそう。リストは長いけど，プレイヤー2のルーチンもプレイヤー1のルーチンとほぼ一緒です。“ほぼ”というのは，選択対象がプレイヤー1では行，プレイヤー2では列になっているところが違っているんですね。

今月のメインルーチン

今回のメインはこんなふうになりました。

*

メイン：　initscrn( )の呼び出し
　plyselect( )の呼び出し
　isover( )が0の間(ゲームオーバーでないということ)，プレイヤー1，2の順に手を打たせる
　終わったらどちらの勝ちかを表示する

plyselect( )：player1を人にする
　player2を人またはコンピュータにセレクト
　先手(先にプレイする側)を決めさせる

*

だいたいわかりましたか？ わかりやすく図示すると，

```
メイン──画面の初期化
      ├─プレイヤーセレクト
      └─打つ手のセレクト┬─手を打たせる
                      └─手をチェック
```

という感じになるんですね。

さて，今月までは表示とかだけだから簡単でしたよね。せっかくリストもあることだし，プレイヤー1，2両方とも人間でやってみてルールを把握してみてくださいね。

来月からはいよいよコンピュータ思考ルーチンの解説に入ります。心の準備はいいでしょうか？ ルールは理解できましたか，プログラムは動いていますか？

では，来月また会おう。ふはははは（と不敵な笑いを残して去る）。

リスト

```
1000 /*選択二十五
1010 /*                          by（で）1991
1020 /*                          Special Thanks to荻窪sanなのだ
1030 /*
1040 /*   メインルーチン  */
1050    /*変数宣言*/
1060   dim int  ary(24)          /*枠の中身*/
1070   dim int  tfrary(24)       /*引数用の配列*/
1080   int      msgy=26          /*メッセージ出力y座標*/
1090   int      ply(2)           /*プレイヤー1，2は人(1)or X68(2)?*/
1100   int      scr1=0,scr2=0    /*プレイヤー1，2のスコア*/
1110   int      msx,msy          /*マウス座標*/
1120   int      usedsel=0        /*すでに印のついている駒の個数*/
1130   int      cplayer          /*現在のプレイヤー*/
1140   int      lowcolum         /*今の行*/
1150   int      depth = 1        /*コンピュータの思考の深さ*/
1160   str      strbuf           /*文字列が一時入るバッファ*/
1170   str      winner           /*ゲーム終了時に勝者が入る*/
1180 /*
1190 /*------  ここからプログラム  ----------
1200 /*
1210   initscrn()
1220   plyselect()
1230   lowcolum=int(rnd()*5)        /*最初の行（列）を決める*/
1240   /*メインループ*/
1250   while(isover(cplayer,lowcolum)=0)
1260     if cplayer=1 then{
1270             /*プレイヤー1は人だけ*/
1280             lowcolum=msel1(lowcolum)
1290     }else{
1300             /*プレイヤー2も今月は人だけね*/
1310             lowcolum=msel2(lowcolum)
1380     }
1390     cplayer=3-cplayer
1400     drawscore()
1410   endwhile
1420 /*ゲームが終わった！ で、どっちが勝ったの?*/
1430   locate 30,msgy:print "     ゲームがおわりましたっ！"
1440   locate 25,msgy+1:print scr1;"対";scr2;"で";
1450   if scr1=scr2 then  print "ひきわけでした！ じゃんじゃん"
1460   if scr1>scr2 then  print "プレイヤー1さんが勝ちました！ おめでとう！"
1470   if scr1<scr2 then  if ply(2)=1 then{
1480                      print "プレイヤー2さんの勝ちです！  おめでとう！"
1490                      }else{
1500                      print "X68000の勝ちです。また挑戦してね。"
1510                      }
1520   end
1530 /* 画面の初期化サブルーチン*/
1540 func    initscrn()
1550   /*画面を消す*/
1560    width 96:screen 2,0,1,1
1570   /*その他の描画*/
1580   drawetc()
1590   /*マス目を描く*/
1600   drawbox()
1610   /*数を適当に配置*/
1620   setnumbers()
1630   drawnumbers()
1640 endfunc
1650 /*マスを描くルーチン*/
1660 func drawbox()
1670  int x,y
1680  for x=0 to 4
1690    for y=0 to 4
1700      box(x*64+96,y*64+64,x*64+96+63,y*64+64+63,15)
1710    next
1720  next
1730 endfunc
1740 /*乱数でマスを埋める*/
1750 func setnumbers()
1760   int i,j,tbl(24)
1770   locate 36,26:print"しばらくおまちください"
1780   /*乱数の使用表を作る*/
1790   for i=0 to 24:tbl(i)=1:next
1800   /*乱数を決める*/
1810   for i=0 to 24
1820     repeat
1830       ary(i)=int(rnd()*25)
1840     until(tbl(ary(i))<>0)
1850     tbl(ary(i))=0
1860   next
1870 endfunc
1880 /*ナンバーを画面に書く*/
1890 func drawnumbers()
1900   int x,y,i
1910   for x=0 to 4
1920     for y=0 to 4
1930       locprn(x,y,15)
1940     next
1950   next
1960 endfunc
1970 /*画面のかざり*/
1980 func drawetc()
1990   symbol(148,0,"★   選択二十五   ★",2,2,2,15,0)
2000   drawscore()
2010 endfunc
2020 /*スコアを書く*/
2030 func drawscore()
2040   locate 64,9:print"PLAYER1"
2050   locate 80,9:print scr1
2060   locate 64,15:print"PLAYER2"
2070   locate 80,15:print scr2
2080 endfunc
2090 /*プレイヤーセレクト*/
2100 func plyselect()
2110  str strbuf
2120  locate 30,msgy:print "    プレイヤー1は人です。          ";
2130  strbuf=inkey$:ply(1)=1
2140  locate 30,msgy:print "      プレイヤー2も人です。          ";
2150  repeat
2160   strbuf=inkey$:ply(2)=1
2170  until(ply(2)=1 or ply(2)=2)
2180  repeat
2190   locate 30,msgy:print " 先手はプレイヤー1,2 どちらですか?  ";
2200   strbuf=inkey$:if strbuf="1" or strbuf="2" then cplayer=v
al(strbuf):print strbuf
2210  until(cplayer=1 or cplayer=2)
2220 endfunc
2230 /*ゲームオーバー判定*/
2240 func isover(cplayer,lowcolum)
2250  int i
2260  int sum=0
2270  /*今のプレイヤーにさす手がなかったらゲームは終わり*/
2280  for i=0 to 4
2290   if cplayer=1 then sum=sum+ary(lowcolum*5+i) else sum=sum
+ary(i*5+lowcolum)
2300  next
2310  if sum=-50000 then return(1) else return(0)
2320 endfunc
2330   /*座標つきの数字プリント*/
2340 func locprn(locx,locy,col)
2350   i=ary(locy*5+locx)-9:if i>=0 and i<=9 then{
2360                          symbol(locx*64+95+16,locy*64+64,st
r$(i),2,2,2,col,0)
2370                          }else if i>-10000 then{
2380                          symbol(locx*64+95,locy*64+64,str$(
i),2,2,2,col,0)
2390                          }else symbol(locx*64+96,locy*64+64
,"x",3,3,2,5,0)
2400 endfunc
2410 /*今セレクトできる行の表示（させない手は表示しない）*/
2420 func linecol(player,lowcolum,r_flg)
2430   int i,col
2440   if r_flg=0 then col = 12 else col =15
2450   if player=1 then{
2460   /*プレイヤー1の場合は縦に*/
2470                 for i=0 to 4
2480                  locprn(i,lowcolum,col)
2490                 next
2500                 }else{
2510   /*プレイヤー2の場合は横に*/
2520                      for i=0 to 4
2530                        locprn(lowcolum,i,col)
2540                      next
2550                      }
2560 endfunc
2570 /*プレイヤー1のさし手セレクトルーチン*/
2580 /*(gotoが使われているのでrenumするときは注意)*/
2590 func msel1(lowcolum)
2600  int nsel                 /*させる手の先頭が入る変数*/
2610  int psel=0               /*今さしているのはいくつ目の数字か、が入る変数*/
2620  int selected=0           /*もうさしたかどうかのフラグ*/
2630  nsel = 5*lowcolum
2640  /*先頭がさせない手の可能性もあるのでさせるなかでいちばん早いものにカーソルを移す*/
2650  while((ary(nsel+psel))=-10000)
2660   psel=psel+1:if psel>4 then psel=0
2670  endwhile
2680  /*プレイヤーの選択範囲の列を色を変えて表示*/
2690  linecol(1,lowcolum,0)
2700  /*カーソルの表示*/
2710  locprn(psel,lowcolum,3)
2720  while(selected<>1)
2730   strbuf=inkey$
2740   /*リターンキーが押されたら点数を増やして、そのマスをもうさせなくして×を書く*/
2750   if strbuf=chr$(13) then selected=1:scr1=ary(nsel+psel)+s
cr1-9:ary(nsel+psel)=-10000:locprn(psel,lowcolum,12):goto 2830
2760   /*そうでない場合は次のマスへカーソルを移す*/
2770     locprn(psel,lowcolum,12)
2780     psel=psel+1
2790     if psel>4 then psel=-1:goto 2780
2800   /*次のマスが打てない手ならその次にカーソルを移す*/
2810   if ary(nsel+psel)=-10000 then goto 2780
2820     locprn(psel,lowcolum,3)
2830  endwhile
2840  linecol(1,lowcolum,1)
2850  return(psel)
2860 endfunc
2870 /*プレイヤー2（人）のさし手セレクト。基本的に1と同じなので注釈は略*/
2880 func msel2(lowcolum)
2890  int nsel,csel
2900  int psel=0
2910  int selected=0
2920  nsel = lowcolum
2930  while((ary(nsel+psel*5))=-10000)
2940    psel=psel+1:if psel>4 then psel=0
2950  endwhile
2960  linecol(2,lowcolum,0)
2970  locprn(lowcolum,psel,5)
2980  while(selected<>1)
2990    strbuf=inkey$
3000    if strbuf=chr$(13) then selected=1:scr2=ary(nsel+psel*5
)+scr2-9:ary(nsel+psel*5)=-10000:locprn(lowcolum,psel,12):goto 3
060
3010        locprn(lowcolum,psel,12)
3020        psel=psel+1
3030        if psel>4 then psel=-1:goto 3020
3040        if ary(nsel+psel*5)=-10000 then goto 3020
3050    locprn(lowcolum,psel,5)
3060  endwhile
3070  linecol(2,lowcolum,1)
3080  return(psel)
3090 endfunc
```

X1用シューティングゲーム

DEFEAT 2

Asano Hidehumi
浅野 英史

前作から２年！ X1/turbo用の横スクロール型シューティングゲームをお届けします。シューティングのエッセンスだけを抜き出した指に優しい超圧縮リストでも，中身はなかなか本格的。都合により予告してから１カ月ずれこんでしまいましたことをおわびします。

X1ユーザーの皆さん，お待たせしました！ 1989年９月号で発表した，X1用横スクロール型シューティングゲーム“Defeat X”の続編が完成しました！ その名も“DEFEAT 2“(ありがちですね)。

ゲームの基本的な内容は前作と同じく，“非パワーアップ型グラディウス”というところですが，いわゆる“２もの”としての成長はしています。

不満はあるでしょうが，そこのところはプログラムの短さに免じて許してやってください。

内容

初めにいっておきますが，このゲームはジョイスティックが必要です。

ゲームの内容は，とにかく撃って撃って撃ちまくって，各面の最後に出てくる敵戦艦の中心部に弾を撃ち込み，破壊してゆくというものです。面数は全部で６面です。

パワーアップなどはなにひとつ存在しません。誰がなんといおうとありません。

自機は６つの装備 (Defeat Xとまったく同じ) を初めから持っており，ジョイスティックのBボタンでいつでもその装備を変更できます。その６つの装備を臨機応変に使い分けて敵を倒していくわけです。

もちろんジョイスティックの上下左右は自機をその方向に動かし，Aボタンは弾の発射です。ちなみに自動連射つきです。

自機が死ぬのは，敵の弾に当たったときと，敵に体当たりしたときと，壁に正面からぶつかったときです。上下と後ろからは壁に当たっても死にはしません。

スコアは敵に１発弾を当てるたびに10点，敵を破壊すると100点が加算されます。２万，５万および８万点で自機が１機増えます。

タイトル画面で‘C’キーを押すとコンティニューできます。回数に制限はありません。また，ゲーム中のポーズはESCキーで，解除はジョイスティックのトリガです。

なお，このゲームはまったくのパターンゲームというわけではありません。それがどういう意味かは何回かプレイすればわかるでしょう。

前作に比べてかなり難しくなっているので，６つの装備をフルに使えるようにならないと最終面はクリアできないでしょう。頑張ってください。

入力方法

このゲームは鬼のよーなデータ圧縮をしており，その展開はBASICで行っています。

展開には，ノーマルBASICかturbo BASIC(どちらもディスク版)が必要です。

また，実行可能なファイルにするために，$4E00_H$～$F7FF_H$までを使えるシステム (S-OSなど) も必要となります。ない人は，後述の方法で，専用のゲームディスクを１枚作る必要があります。

それでは打ち込み方です。まず最初に，リスト２ (ALLROUND) とリスト３ (MAIN) をMACINTO-Cなどのマシン語入力ツールを使って打ち込みセーブします。

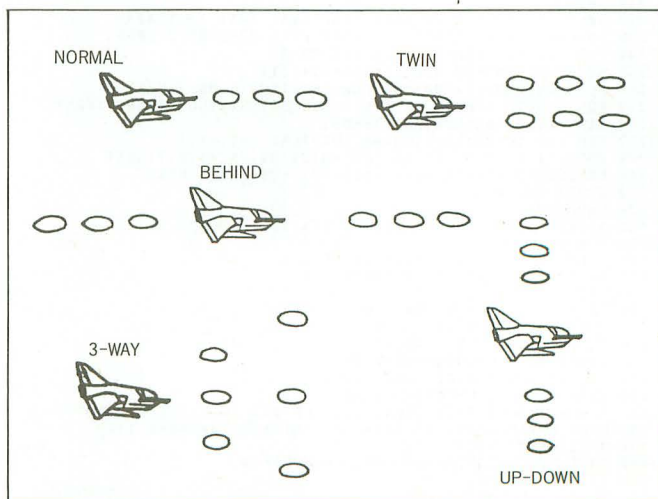
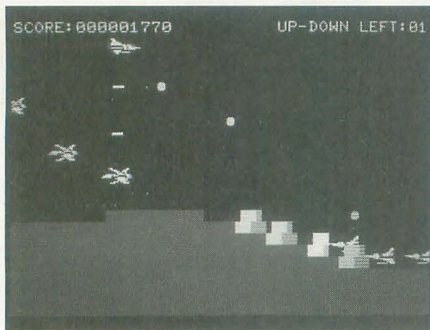
次に，リスト１ (EXTRACT) をBASICから打ち込みセーブします。そしてリスト１をRUNすると，自動的にデータを展開していきます。30分ほどかかりますので注意してください。また，リスト１は初めにリスト２を読むので，不都合がないようにお願いします (なにがいいたいかわかりますよね)。

リスト２の実行が終わると，ディスク(テープ) 上に“DATA.”,“PCGDATA.”,“ROUND n.”{n:1-6} という８つのファイルが生成されます。

そして，S-OSなどのシステム上で，その７つのファイルとリスト３を，次に示すアドレスに読み込んでください。

MAIN.	：$4E00_H$
DATA.	：6500_H
ROUND1.	：8000_H
ROUND2.	：9000_H
ROUND3.	：$A000_H$
ROUND4.	：$B000_H$
ROUND5.	：$C000_H$
ROUND6.	：$D000_H$
PCGDATA	：$E000_H$

そして最後に$4E00_H$～$F7FF_H$までをディスクにセーブしてください。これで実行ファイルができました。あとは$4E00_H$番地をコールすればゲームが始まります。

なお，もし万一，皆さんがS-OSなどを持っていなかったり，このゲームをIPL起動できるようにしたいなどという寛大な考え

武器のバリエーション

をなさったときのために，IPL起動ディスク作成プログラムを用意してあります。

まずリスト2を展開して，上述の8つのファイルを作ります。

そしてBASIC上でリスト4（MAKEDATA）を打ち込み，ドライブ0に上述のファイルの入ったディスクを，ドライブ1にはフォーマットずみのディスクを入れてRUNします。しばらく待てばIPL起動ができるディスクができあがるでしょう。

最後に

今回はBGMなしなので，本当ならもっと短くなってもいいはずなわけですが，なんだかんだといっているうちに，かなり膨らんでしまいました。どうもすみません。

その埋め合わせといってはなんですが，ゲーム性にはけっこう気をつかっており，心の中ではいままでのX1/turbo用のどんなゲームにもひけをとらない面白さだと自負しています。あくまで心の中でですが。

エンディングはすごく手抜きに見えますが，これは「いかに短く，かつ見れるものにするか」という努力の結果なのです（実際4日はかかっている）。どうか誤解のないように。

ところで，先にも書きましたが，このゲームはデータ部を圧縮して，その部分だけを別にしてあります。PCGのデータはランレングス法で圧縮してありますし，BASICの関数を利用してテーブルを作ったりもしています。

特にえげつないのはマップデータの圧縮です。こんな方法でマップデータを圧縮したのは，たぶん日本で，いや世界でも僕ひとりでしょう。興味のある人は展開中にCtrl-0（フルキーのほう）を押してみましょう。きっと笑えますよ。

というわけで，データの圧縮率はまあまあなわけですが，そのために必要なリスト1はけっこう大きいです。

せっかく打ち込んでも1回しか使わないなんてもったいない！　という方もいるでしょうが，リスト1に別のデータを食わせてやれば，違う面を楽しむことができるのです。そう，「追加面」というやつです。

もし皆さんがこのゲームに好意的な評価をくだしてくだされば追加面の発表も可能かもしれません（というわけで，STUDIO Xのほうへヨロシクお願いします）。

なお，次回作はシューティングゲームにはならない予定です。あしからず。

参考文献
試験に出るX1，祝　一平，ソフトバンク

リスト1　EXTRACT

```
10 OF=&H1000  :' turbo = 0
20 CLEAR &HC7FF+OF
30 DIM CH(4),MM(40)
40 DIM X(100),Y(100)
50 ADR=&HE000+OF:LOADM"ALLROUND",ADR
60 WIDTH 80 :' TURBO --WIDTH 80,25,0
70 SCREEN
80 '-------
85 FOR I=32 TO 90:DEFCHR$(I)=STRING$(3,LEFT$(CGPAT$(I),8)):NEXT
90 PRINT"Now making PCGDATA ...."
100 O$=STRING$(8,CHR$(0))
110 FOR I=0 TO 7
120   I$=CHR$(0,0,0,0,2^I,0,0,0)
130   DEFCHR$(I   )= I$+ O$+ O$
140   DEFCHR$(I+8 )= I$+ O$+ I$
150   DEFCHR$(I+16)= I$+ I$+ I$
160 NEXT
170 FOR I=0 TO 3
180   I$=CHR$(0,0,0,3*4^I,3*4^I,0,0,0)
190   DEFCHR$(I+24)= O$+ I$+ O$
200   DEFCHR$(I+28)= O$+ I$+ I$
210 NEXT
220 A$="0000 8000 8020 A020 A0A0 A4A0 A4A1 A5A1 A5A5"
230 A$= A$+" E5A5 E5B5 F5B5 F5F5 F7F5 F7FD FFFD FFFF"
240 FOR I=0 TO 16:B$="":FOR J=0 TO 3
250 B$=B$+STRING$(2,MID$(A$,I*5+J+1,1))
260 NEXT:DEFCHR$(I+200)=HEXCHR$(STRING$(6,B$)):NEXT
270 FOR I=0 TO 3:READ A$:A$(I)=HEXCHR$(STRING$(4,A$)):NEXT
280 READ A$,B$:A$=HEXCHR$(A$+B$)
290 FOR I=0 TO 25:B$=BIN$(ASC(MID$(A$,I+1,1)))
300 FOR J=1 TO 3:A(J)=VAL("&B"+MID$(B$,J*2+1,2)):NEXT
310 DEFCHR$(32+I)=A$(A(1))+A$(A(2))+A$(A(3)):NEXT
320 FC=91:CN=39
330 X=0:Y=0
340 A=PEEK(ADR):L=A AND 31:C=INT(A/32)
350 FOR J=1 TO L
360 PSET(X,Y,C):PSET(X+1,Y,C) :X=X+2
370 IF X>(CN*8-1)*2 THEN X=0:Y=Y+1
380 NEXT
390 IF Y<>8 THEN ADR=ADR+1:GOTO 340
400 FOR I=0 TO CN*2-1   :A$=""
410 FOR J=0 TO 2:FOR K=0 TO 7
420 P=&H4000+I+J*&H4000+K*2048
430 A$=A$+CHR$(INP(P)):OUT P,&HFF
440 NEXT:NEXT:DEFCHR$(I+FC)=A$: NEXT
450 FOR I=0 TO 255:A$=CGPAT$(I):FOR J=0 TO 7:FOR K=0 TO 2
460 POKE OF+&HC800+I*24+J*3+K,ASC(MID$(A$,J+K*8+9,1))
470 NEXT:NEXT: NEXT
480 SAVEM"PCGDATA",OF+&HC800,OF+&HDFFF,0
490 '-------
500 PRINT"Now making DATA ...."
510 P=0:C=0 :ADR2=OF+&HCD00 :ADR=ADR+1
520 IF P>0 THEN POKE ADR2,C+32:P=P-1:ADR2=ADR2+1: GOTO 520
530 P=PEEK(ADR) :ADR=ADR+1
540 IF P<>255 THEN C=INT(P/16):P=(P AND 15)+1:GOTO 520
550 FOR D=0 TO 2:READ A$
560 FOR J=1 TO 7
570 FOR I=1 TO LEN(A$)
580 B$=RIGHT$("00000000"+BIN$(ASC(MID$(CGPAT$(ASC(MID$(A$,I,1)))
,J,1))),8)
590 FOR K=1 TO 8:POKE ADR2,ASC(MID$(B$,K,1))-16:ADR2=ADR2+1:NEXT
600 NEXT:NEXT :NEXT
610 FOR I=0 TO 63 : K=RAD(I*360/64)
620 X=INT(COS(K)*20):X=X-(X<0)*256
630 Y=INT(SIN(K)*10):Y=Y-(Y<0)*256
640 POKE ADR2,X,Y:ADR2=ADR2+2
650 NEXT
660 FOR I=0 TO 7:READ D$
670 MEM$(ADR2,LEN(D$))=D$:ADR2=ADR2+LEN(D$)
680 NEXT
690 '-------
700 FOR I=0 TO 4:FOR J=0 TO 5:P=I*12+J
710 POKE &HD800+OF+P , 97+J+I*6
720 POKE &HD806+OF+P ,133+J+I*6
730 NEXT :NEXT
740 FOR I=0 TO 4:"PEEKWORD":CH(I)=PP:NEXT
750 FOR I=0 TO 15:"PEEKWORD":MM(I)=PP:MM(I+16)=PP+640:NEXT
760 ADR2=&HD880+OF:FOR I=0 TO 31
770 P1=PEEK(ADR):P2=PEEK(ADR+1)
780 POKE ADR2    ,255-INT(P1/64)*43 ,P2   ,INT((P1 AND 63)/16),P
1 AND 15
790 POKE ADR2+256,255-INT(P1/64)*43 ,P2+32,INT((P1 AND 63)/16),P
1 AND 15
800 ADR2=ADR2+4
810 P1=PEEK(ADR+2):PP=MM( INT(P1/8)    ):"POKEWORD"
820 ADR2=ADR2+256-2 :PP=MM( INT(P1/8)+16 ):"POKEWORD" :ADR2=ADR2
-256
830 PP=CH( P1 AND 7 ):"POKEWORD"
840 ADR2=ADR2+256-2     :"POKEWORD" :ADR2=ADR2-256
850 ADR=ADR+3: NEXT
860 ADR2=&HDB00+OF:FOR I=0 TO 15
870 P1=PEEK(ADR):P2=PEEK(ADR+1)
880 DX=(248+INT(P1/16)) AND 255
890 DY=(248+(P1 AND 15)) AND 255:DDY=(256-DY) AND 255
900 POKE ADR2   ,DX,DY ,INT(P2/8),P2 AND 7
910 POKE ADR2+640,DX,DDY,INT(P2/8),P2 AND 7 : ADR2=ADR2+4 :ADR=A
DR+2
920 IF P1<>136 OR P2<>0 THEN  870
930 NEXT:SAVEM "DATA",&HCD00+OF,&HDFFF+OF,0
940 '-------
950 FOR R=1 TO 6:P$=HEX$(R)
960 PRINT"Now making ROUND"+P$+"...."
970 GOSUB "MKMAP"
980 GOSUB "EVENT"
```

```
990 GOSUB "BOSS"
1000 FOR I=0 TO &HCFF:POKE I+&HD000+OF ,INP(I+&H4000):NEXT
1010 SAVEM "ROUND"+P$,&HD000+OF ,&HDFFF+OF
1020 CLS0:ADR=ADR+1:BEEP:NEXT
1030 BEEP:BEEP:BEEP:  END
1040 '-------
1050 LABEL "MKMAP"
1060 CLS0
1070 LINE(0,0)-(511,25),PSET,6,B
1080 F=PEEK(ADR):ADR=ADR+1
1090 PT=PEEK(ADR)+1
1100 FOR I=0 TO PT-1
1110 X(I)=PEEK(ADR+I*2+1)
1120 Y(I)=PEEK(ADR+I*2+2)
1130 NEXT:ADR=ADR+PT*2+1 :  GOSUB "REDRAW"
1140 VADR=&H4000 :CLS 1
1150 FOR X=1 TO 510
1160 FOR Y=0 TO 2
1170 N$="" :FOR I=0 TO 7
1180 P=POINT(X,Y*8+ I+1  )
1190 IF P>5 THEN N$="1"+N$ ELSE N$="0"+N$
1200 NEXT:OUT (VADR+4),VAL("&B"+N$):VADR=VADR+1
1210 NEXT:NEXT:OUT (&H4000),F:RETURN
1220 '-------
1230 LABEL "REDRAW"
1240 LINE (1,1)-(510,24),PSET,0,BF
1250 PX=0:PY=100
1260 FOR I=0 TO PT-1
1270 IF Y(I)>=200 AND Y(I)<>255 THEN PAINT(X(I)*2,Y(I)-200),7,6
:GOTO 1310
1280 IF X(I)=255 AND Y(I)=255 THEN PY=100:GOTO 1310
1290 IF PY=100 THEN PX=X(I):PY=Y(I)       :GOTO 1310
1300 LINE (PX*2,PY)-(X(I)*2,Y(I)),PSET,7 :PX=X(I):PY=Y(I)
1310 NEXT:RETURN
1320 '-------
1330 LABEL"EVENT
1340 I=0:J=0:ADR2=&HD000+OF
1350 P=PEEK(ADR+J)
1360 IF P >191 THEN POKE ADR2+I,3:ADR=ADR+J+1:GOTO 1410
1370 IF P <64  THEN POKE ADR2+I,0,P :J=J+1:I=I+2:GOTO 1350
1380 Q=PEEK(ADR+J+1)
1390 POKE ADR2+I,INT(P/64),P AND 63,INT(Q/8),Q AND 7
1400 J=J+2 :I=I+4 :GOTO 1350
1410 FOR J=0 TO I:OUT (&H4000+J+&H600),PEEK(ADR2+J):NEXT
1420 RETURN
1430 '-------
1440 LABEL"BOSS"
1450 VADR=&H4900
1460 DX=PEEK(ADR)*2:DY=PEEK(ADR+1)
1470 OUT VADR,PEEK(ADR+2)
1480 FOR I=0 TO 5:OUT VADR+I*2+3,PEEK(ADR+I+3) :NEXT
1490 OUT VADR+17,DY:OUT VADR+18,DX
1500 VADR=&H4A00
1510 TBL$=" 025!'()&%-"+CHR$(34)+"."
1520 ADR=ADR+9  :P=0
1530 FOR Y=0 TO INT(DY/2)  :FOR X=0 TO DX-1
1540 IF P<>0 THEN 1560
1550 P=PEEK(ADR):CH=ASC(MID$(TBL$,INT(P/16)+1,1)):P=(P AND 15)*2
:ADR=ADR+1
1560 OUT VADR+    Y   *DX+X ,CH
1570 OUT VADR+(DY-Y-1)*DX+X ,CH
1580 P=P-1:NEXT:NEXT
1590 RETURN
1600 '-------
1610 LABEL"POKEWORD"
1620 PP$=RIGHT$("0000"+HEX$(PP),4)
1630 MEM$(ADR2,2)=HEXCHR$(RIGHT$(PP$,2)+LEFT$(PP$,2)) :ADR2=ADR2
+2
1640 RETURN
1650 '-------
1660 LABEL"PEEKWORD"
1670 PP=PEEK(ADR)+PEEK(ADR+1)*256:ADR=ADR+2
1680 RETURN
1690 '-------
1700 DATA "0000","55AA","AA55","FFFF"
1710 DATA"00B08CBC 83B38FBF 90849481 91859598"
1720 DATA"B892B296 B6AC86A6 8EAE96B6 "
1730 DATA "GAMEOVER","MISSION","COMPLETE"
1735 DATA "<<< PRESENTED BY H:ASANO >>>"
1740 DATA "PRESS TRIGGER TO START!"
1750 DATA "SCORE:000000000"
1760 DATA "LEFT:00"
1770 DATA "NORMAL  TWIN    UP-DOWN 3-WAY   BEHIND  "
1780 DATA "YOU DEFEATED ALL OF ENEMY BATTLESHIP"
1790 DATA "    THIS MISSION WAS COMPLETED       "
1800 DATA " &&&- &&&---&&&&-- &&--  '''. '''...''''.. ''.. "
```

リスト2 ALLROUND

```
E000 1F 1F 1F 1F 14 41 0A E3 : BE
E008 01 E2 44 06 E9 0B 41 06 : 68
E010 26 01 A1 28 41 02 E7 02 : 1C
E018 A1 E6 41 06 EB 0B 42 02 : 08
E020 A4 E3 01 E3 06 E2 42 1F : B4
E028 1F 1F 1F 1F 0E 41 01 E9 : B5
E030 01 E2 01 E9 07 42 E2 01 : F9
E038 EA 01 41 0E A2 26 05 42 : 49
E040 E3 01 A2 E5 44 09 E4 02 : 9E
E048 EB 03 E7 05 E5 02 E2 41 : E4
E050 0B C4 1F 18 E2 16 41 21 : 60
E058 1F 0B E2 1F 0A E2 01 E5 : FD
E060 13 E8 11 A2 24 0D A2 E4 : 65
E068 43 12 E8 0D E9 0D C6 02 : 08
E070 46 03 E3 1F 08 E3 14 41 : 8B
E078 23 15 E3 12 E1 42 0A E3 : 3D
---------------------------------
SUM: 4C B2 F0 4D F1 26 2C 8B 6075

E080 0A E2 07 E6 1F 01 E6 06 : E5
E088 42 28 02 A3 01 22 07 A4 : DD
E090 02 A2 E4 01 A2 12 E5 10 : 32
E098 E5 42 04 E8 01 C6 01 48 : 23
E0A0 03 E3 1F 06 E2 42 12 25 : 66
E0A8 0C E3 05 E2 43 12 E2 42 : 4F
E0B0 0B E4 05 E3 09 E4 01 E6 : AB
E0B8 41 11 A2 E6 02 E4 1F 01 : E0
E0C0 A5 02 A2 E3 01 A1 13 E3 : C4
E0C8 0E E3 02 E4 42 03 E8 01 : 05
E0D0 C6 01 48 03 E5 1F 02 E3 : FB
E0D8 43 08 A2 22 02 29 09 E4 : 27
E0E0 05 E6 10 E8 0C E4 02 E3 : B8
E0E8 0B E2 1F 1F 13 A3 1F 04 : 04
E0F0 E4 06 E4 0C C6 02 46 03 : EB
E0F8 E8 15 A2 E2 02 41 E9 05 : B2
---------------------------------
SUM: 26 7A FF 04 04 CD 3D EA 4ABC

E100 A6 22 0A 41 06 A2 01 E2 : 9E
E108 01 E2 01 E8 0B E3 01 E1 : 9C
E110 08 41 0D E1 01 E2 02 E1 : FD
E118 1F 1F 1F 1F 1F 04 C4 0A : 6D
E120 41 01 EB 01 E2 42 06 E5 : 3D
E128 A4 E3 43 E4 44 01 2B 01 : 1F
E130 2A 42 03 A3 01 E2 0A E2 : E1
E138 07 E2 A4 E2 01 E8 01 42 : 9B
E140 04 A2 E2 01 E2 02 E5 01 : 53
E148 E2 41 1F 1F 1F 1F 1F 0F : CD
E150 2F 2F 0F 06 22 90 05 21 : 4B
E158 91 0F 0F 03 22 90 04 23 : 8B
E160 91 03 22 92 04 93 0F 21 : 0F
E168 91 03 24 92 00 26 92 25 : 27
E170 93 00 23 93 03 93 05 21 : 05
E178 91 03 24 92 00 22 01 21 : 8E
---------------------------------
SUM: D0 96 B8 05 A5 27 B8 94 AC71

E180 92 22 01 20 93 00 23 93 : 1E
E188 03 93 05 21 91 03 21 01 : 72
E190 20 92 00 20 05 92 20 05 : 8E
E198 92 00 21 01 93 03 93 07 : E4
E1A0 91 09 91 09 90 08 90 06 : 62
E1A8 91 05 91 04 2B 0F 0F 0F : 83
E1B0 01 21 07 21 0C 23 0F 0E : 96
E1B8 21 09 21 09 21 0F 0F 00 : 93
E1C0 21 0A 21 07 27 0F 05 21 : AF
E1C8 04 21 09 21 02 23 03 21 : 98
E1D0 05 23 05 23 03 25 01 21 : 9A
E1D8 09 21 01 27 01 21 03 27 : 9E
E1E0 01 21 03 21 03 21 03 21 : 8E
E1E8 06 22 03 21 07 21 03 21 : 98
E1F0 07 21 03 21 03 21 01 2A : 9B
E1F8 08 23 03 21 05 23 05 23 : 9F
---------------------------------
SUM: D4 75 AD 8F E3 DF CC DC 4E2F

E200 01 21 01 21 04 2A 04 21 : 97
E208 02 21 06 21 02 21 05 21 : 93
E210 02 21 06 21 02 21 05 21 : 93
E218 02 21 06 21 02 21 05 21 : 93
E220 02 21 06 21 02 21 04 2A : 9B
E228 06 52 00 71 51 01 71 51 : DD
E230 02 71 51 02 71 51 03 71 : FC
E238 51 05 52 00 23 00 23 91 : 7F
E240 00 93 FF 00 70 0C 70 18 : 96
E248 70 24 70 30 70 00 73 08 : 1F
E250 73 14 73 38 73 5C 73 94 : 08
E258 73 9C 73 AC 73 C4 73 F8 : D0
E260 73 18 74 44 74 74 74 8C : 2B
E268 74 AC 74 F4 74 01 00 00 : FD
E270 C1 00 08 C1 00 10 02 03 : 9F
E278 19 C2 03 21 01 05 2A C1 : F0
---------------------------------
SUM: 79 5A 04 46 A0 B6 17 FD 7D51

E280 05 32 C1 05 3A 02 08 43 : 84
E288 C2 08 4B C2 08 53 02 0B : 3F
E290 5C C2 0B 64 02 0D 68 C2 : C6
E298 0D 70 01 0F 79 14 00 02 : 1C
E2A0 D3 10 0A 11 12 1B 11 13 : 4F
E2A8 2C 21 14 00 E1 14 08 E1 : 3F
E2B0 14 10 21 17 29 E1 17 31 : AE
E2B8 31 19 00 F1 19 08 F1 19 : 66
E2C0 10 31 1C 1A 31 1D 2B 00 : F0
E2C8 00 00 00 00 00 48 18 88 : E8
E2D0 00 48 B8 C8 F1 88 00 48 : 89
E2D8 F0 49 19 6A 19 8A 19 AA : 22
E2E0 19 C9 19 C8 F1 C8 F1 88 : F5
E2E8 00 48 21 49 19 5A 19 49 : 87
E2F0 19 48 21 47 19 56 19 47 : 98
E2F8 19 88 00 48 21 49 19 5A : C6
---------------------------------
SUM: BF 69 9F 3F 71 C6 2B 3C 9987

E300 19 49 19 48 21 47 19 56 : 9A
E308 19 47 19 48 21 49 19 8A : CE
E310 19 D9 19 D8 F1 88 00 28 : 84
E318 18 88 00 28 58 8B 22 48 : 15
E320 20 88 00 28 C8 8A 29 BA : 05
E328 21 D8 F1 D8 F1 88 00 48 : 83
E330 79 49 11 6A 11 8A 11 AA : 93
E338 11 C9 11 C8 19 C7 11 A6 : 4A
E340 11 86 11 66 11 47 11 88 : FF
E348 00 48 79 49 11 6A 11 8A : 20
E350 11 69 19 48 21 28 F1 88 : 9D
E358 00 48 51 49 11 6A 11 8A : F8
E360 11 59 19 48 29 28 59 8A : FF
E368 19 C8 F1 C8 F1 88 00 3A : 4D
E370 20 39 20 28 50 48 38 47 : B8
E378 18 76 18 96 18 C7 18 C8 : FB
---------------------------------
SUM: B2 B2 94 D0 44 08 6C 99 9084

E380 B0 C9 28 CA A0 88 00 3A : CD
E388 20 39 20 28 70 37 20 36 : 9E
E390 A0 88 00 48 42 59 12 7A : 97
E398 32 59 1A 48 3A 38 2A 37 : C0
E3A0 F2 88 00 48 42 59 12 7A : E9
E3A8 32 59 1A 48 52 47 1A 66 : 06
E3B0 12 86 12 A6 12 C7 1A C8 : 0B
E3B8 1A C9 1A AA 12 8A 12 6A : BF
E3C0 12 49 1A 38 F2 88 00 48 : 6F
E3C8 49 59 11 7A 39 78 1B 76 : 6F
E3D0 39 57 11 88 00 2B 23 04 : 7B
E3D8 19 15 16 2B 18 32 16 36 : 05
E3E0 0D 39 0B 3B 15 4A 17 5B : 5D
E3E8 16 62 12 6B 13 7B 18 83 : 1E
E3F0 11 8B 18 94 16 97 11 94 : 9A
E3F8 0F 95 0C 9D 0B A1 10 9F : A8
---------------------------------
SUM: E2 DC 3B 9E D0 9B 58 3C 4974

E400 14 A9 17 BA 16 CD 18 D9 : 62
E408 15 DC 0F E0 16 E3 16 E9 : D8
E410 12 F2 18 FF 19 EA DF 9C : 99
E418 DF 83 DF 38 DF 16 E0 3D : 8B
E420 80 C4 01 50 34 19 43 54 : 79
E428 0D 80 C4 09 80 C3 19 46 : FC
E430 35 21 68 A4 3D 80 B4 05 : D8
E438 80 B3 07 80 BC 15 45 43 : 13
E440 29 80 C3 05 80 C3 05 69 : 22
E448 95 3F 01 5C 36 29 80 AB : BB
```

```
E450 09 4C 45 01 80 B4 0B 80 : 5A
E458 BC 35 51 3C 23 66 8D 03 : 97
E460 80 CC 05 80 CC 1B 80 BB : F3
E468 07 48 34 2F 80 B5 05 80 : 6C
E470 B4 05 80 BB 03 44 33 33 : A1
E478 68 94 11 47 45 3B 6E C4 : 06
---------------------------------
SUM: 82 FF 75 9D BE 76 85 46 2488

E480 19 80 CC 05 80 CC 17 48 : 15
E488 44 31 80 BC 05 80 BC 0F : 01
E490 5A 36 3F 0F C0 12 0D 32 : EF
E498 00 28 19 00 00 00 07 56 : 9E
E4A0 09 C2 59 06 BC 52 01 B4 : ED
E4A8 03 C5 02 B3 0A C4 03 B2 : 00
E4B0 0A B3 51 B1 0C B1 82 B2 : B0
E4B8 00 FF 29 32 02 19 08 15 : 92
E4C0 12 17 1A 17 26 13 33 17 : DD
E4C8 43 16 4A 12 58 10 60 15 : 92
E4D0 75 17 86 16 97 17 A9 15 : 94
E4D8 B5 17 C2 15 CD 17 E0 15 : 7C
E4E0 E6 17 ED 15 F1 17 FE 19 : 1E
E4E8 EB E0 FF FF 22 0B 2C 09 : 2B
E4F0 3B 0A 42 0E 35 10 2C 0D : 13
E4F8 22 0B 30 D3 FF FF 65 0B : 9E
---------------------------------
SUM: 7A AF 83 B5 42 C0 4C 9C F428

E500 6C 06 78 07 80 0B 75 0E : FF
E508 73 12 71 0D 65 0B FF FF : 71
E510 6E D1 A9 0A C1 06 CE 0A : 91
E518 C4 0C C1 10 B9 0C A9 0A : 19
E520 B6 D2 3B 80 C3 01 50 3E : 95
E528 05 80 C3 13 5C 46 3F 05 : 41
E530 80 AC 03 46 2D 07 80 B3 : DC
E538 0D 80 BB 13 82 5D 05 82 : C1
E540 5C 0D 81 9C 0D 80 64 0B : 82
E548 81 93 01 80 74 2D 48 35 : B3
E550 15 53 34 27 80 9C 05 80 : 64
E558 A3 07 82 AC 03 4C 4E 01 : 76
E560 80 B4 0D 6D B5 3F 07 83 : 2C
E568 94 07 81 8B 0F 80 54 07 : 91
E570 80 5C 0F 50 2B 0B 50 93 : 54
E578 1F 5C 45 21 4C 4D 05 6C : EB
---------------------------------
SUM: A1 E0 29 72 6C 7F AE E3 0889

E580 7D 27 51 3E 27 66 9D 3F : 9C
E588 03 6F D4 23 83 83 09 83 : FB
E590 7C 01 80 4C 0B 80 52 19 : 3F
E598 45 AE 0B 57 2E 17 51 3D : 28
E5A0 11 51 3D 27 6A AD 0B 4A : 32
E5A8 3D 3D C0 0F 0F 32 14 1E : BC
E5B0 14 0F 00 00 03 64 02 53 : DF
E5B8 09 62 55 02 65 02 53 C4 : 40
E5C0 05 64 C7 06 43 C4 03 55 : 95
E5C8 45 72 B1 08 B3 41 73 B1 : 88
E5D0 05 C2 83 B1 62 00 FF 38 : 94
E5D8 4D 01 19 08 14 0E 15 11 : B7
E5E0 17 23 16 2C 11 36 0F 3B : 0D
E5E8 17 4C 17 4F 15 4D 12 50 : 8D
E5F0 0F 56 10 57 15 5C 17 6D : C1
E5F8 15 7B 17 84 15 88 17 9C : 7B
---------------------------------
SUM: 9A 1D 6A 59 80 3F 96 7A FDE7

E600 16 9E 0F A2 0C A7 0F A8 : CF
E608 17 B4 16 B8 12 B9 0E BB : 2D
E610 0F BE 16 CE 17 D9 16 DE : 95
E618 13 DF 10 E9 10 EA 17 EF : EB
E620 16 FF 18 E4 DF FF FF 01 : EF
E628 01 0D 05 1A 03 28 06 2C : 8A
E630 03 3B 02 44 05 48 02 56 : 29
E638 02 5B 04 65 05 66 02 70 : A3
E640 02 78 06 7B 09 7E 06 80 : 08
E648 03 86 04 88 09 89 0D 90 : 44
E650 0E 96 0B 8D 0A 8D 06 92 : 6B
E658 03 A9 02 AF 06 B7 02 C2 : DE
E660 03 C7 07 CC 06 D4 02 D8 : 51
E668 03 DC 07 DE 0A E9 0A EB : AC
E670 04 FF 01 E7 CA 2F 46 74 : 9E
E678 0F 80 C4 05 82 C4 05 80 : 23
---------------------------------
SUM: 9A F0 58 8D AF F3 C5 3E 998B

E680 C3 01 83 44 0B 81 3C 1B : 6E
E688 69 AD 07 49 6D 27 81 45 : C0
E690 05 83 3B 05 83 3C 0F 4D : E3
E698 3C 17 41 44 0F 82 C4 05 : 32
E6A0 82 C4 11 6D C5 19 83 34 : 59
E6A8 05 83 33 03 47 54 01 81 : DB
E6B0 34 03 80 C4 23 5C 3E 0F : 47
E6B8 82 BC 11 80 C3 01 83 45 : 5B
E6C0 05 82 C4 17 53 4D 03 83 : 88
E6C8 33 05 81 34 0B 6C 94 19 : 11
E6D0 40 5C 19 80 C4 11 82 C4 : 50
E6D8 0F 83 3C 01 4D 4C 03 81 : EC
E6E0 3C 0F 66 B5 1F 82 5A 05 : 66
E6E8 82 5A 01 83 42 07 80 65 : 8E
E6F0 01 81 3D 03 80 65 05 81 : 2D
E6F8 3D 19 4D 64 35 82 C3 09 : 8A
---------------------------------
SUM: 2D B7 66 F5 81 B6 93 90 2501

E700 80 C4 15 57 5D 05 81 44 : D7
E708 07 81 3B 27 82 BC 05 80 : AD
E710 BC 13 6D A4 11 49 64 27 : C5
E718 81 34 05 81 34 11 66 A5 : 8B
E720 3F 05 80 C5 01 81 45 05 : 55
E728 82 BB 01 83 43 03 82 BB : 44
E730 03 83 3B 05 80 BD 03 81 : 87
E738 3D 11 41 4D 0D 41 5D 13 : 9A
E740 C0 11 0F 32 19 2D 14 08 : 74
E748 00 03 0E B3 03 B2 07 B4 : 34
E750 01 B7 03 B5 07 B3 65 B1 : 40
E758 07 64 45 61 05 62 4A 61 : 23
E760 0A B3 72 B1 07 B4 83 B1 : CF
E768 00 FF 21 4F 00 01 0A 03 : 7D
E770 0D 07 11 07 14 04 19 02 : 5F
E778 23 03 28 07 2F 05 32 02 : BD
---------------------------------
SUM: C7 CB F0 46 67 4F 19 6A 4DA4

E780 43 03 48 07 4F 03 53 03 : 3D
E788 58 07 5E 03 70 02 79 05 : B0
E790 7F 02 9D 02 A2 09 A8 0C : 7F
E798 B4 02 BF 03 C3 07 C4 0A : 10
E7A0 CC 0C D4 0A CC 07 CC 02 : 57
E7A8 D8 02 E2 04 E9 09 EE 0A : AA
E7B0 FF 01 F5 C9 FF FF 01 18 : D5
E7B8 0B 15 0E 12 1A 14 21 0E : 9D
E7C0 26 10 2A 16 34 17 39 13 : 0D
E7C8 3E 14 40 17 49 10 4E 15 : 65
E7D0 54 15 58 0F 60 17 69 17 : C7
E7D8 6E 14 66 14 6D 11 72 11 : FD
E7E0 75 0E 7F 0A 85 0A 88 0D : 30
E7E8 80 10 7F 15 8C 17 93 14 : 6E
E7F0 9C 17 A7 15 AB 17 B6 17 : FE
E7F8 BE 15 C2 12 CA 13 D0 16 : 6A
---------------------------------
SUM: F1 C9 4A 8E C2 D2 17 EE 8E99

E800 E0 17 E8 14 E8 11 F0 12 : EE
E808 FF 18 F5 E0 1F 50 3E 0F : A8
E810 47 65 13 80 A7 05 80 A7 : 12
E818 01 83 44 07 83 3C 0D 80 : 1B
E820 AD 03 4C 6C 3F 82 BC 09 : EE
E828 80 BE 03 83 4C 07 53 53 : BD
E830 23 81 35 0D 82 AC 05 47 : 60
E838 4D 1B 51 54 23 80 B5 03 : 68
E840 81 3D 03 80 B5 0B 83 3C : C0
E848 09 53 5B 21 81 3E 15 80 : 2C
E850 C6 0B 82 C4 05 5C 46 01 : BF
E858 82 C2 0F 66 95 05 81 36 : 0A
E860 07 83 34 39 81 34 05 51 : 02
E868 42 01 82 B2 05 83 9A 03 : 9C
E870 82 BB 07 83 94 07 80 BE : A0
E878 01 81 8D 19 6D 9D 23 49 : 9E
---------------------------------
SUM: 62 91 42 1D B8 5C 25 3C 29E1

E880 4D 13 80 C6 03 6C 9D 3F : F1
E888 03 83 5D 0B 51 83 05 82 : 49
E890 C4 09 4B 94 21 5C 54 01 : 7E
E898 82 B4 01 83 3D 11 66 9C : 0A
E8A0 27 83 32 07 83 32 09 80 : 21
E8A8 56 03 81 36 19 4D 5C 09 : DB
E8B0 82 C4 0B 51 55 0B 81 46 : C9
E8B8 0B 67 B5 05 80 B5 3F 0D : AD
E8C0 81 46 01 80 BE 03 51 5C : B6
E8C8 01 83 3B 03 82 C3 0B C0 : D2
E8D0 0F 0D 23 00 1E 14 0F 00 : 80
E8D8 06 05 B4 09 82 B7 05 82 : 88
E8E0 BB 07 64 09 76 B2 02 B8 : 11
E8E8 61 74 B2 06 82 64 00 FF : 72
E8F0 32 2E 10 19 10 11 1D 11 : D8
E8F8 1D 19 2C 19 2C 0E 32 0E : F5
---------------------------------
SUM: A2 A1 01 48 37 61 42 AE D881

E900 32 13 39 13 39 19 45 19 : 41
E908 52 11 58 11 61 19 73 19 : D2
E910 73 13 83 13 83 19 98 19 : 69
E918 98 16 AB 16 AB 12 B3 12 : F1
E920 B3 16 BA 16 BA 19 D1 19 : 56
E928 D1 14 DD 14 DD 19 E4 19 : C9
E930 E4 11 E8 11 E8 19 ED 19 : F5
E938 F1 12 F6 11 FA 13 FC 19 : 2C
E940 F8 DE E6 DE DA DE AF DE : DF
E948 7C DE 59 DE 35 DE 18 DE : 9A
E950 31 51 2D 07 51 2D 07 80 : BB
E958 97 09 82 95 09 82 95 03 : DA
E960 4C 6D 07 80 97 11 5C 2E : 72
E968 25 6D BD 19 49 45 09 80 : 7F
E970 7E 09 82 A6 07 82 A4 09 : E5
E978 80 A7 11 66 75 17 44 3E : AC
---------------------------------
SUM: 93 3A 79 96 06 15 51 F5 D156

E980 19 67 A4 0B 80 B6 0D 50 : C2
E988 2D 0F 80 97 03 53 2D 05 : DB
E990 82 96 07 82 95 0B 4B 65 : F1
E998 01 6B 4D 05 82 B4 21 48 : 5D
E9A0 35 13 51 46 19 6D C5 19 : 43
E9A8 80 A5 0B 82 A5 01 68 8D : 4D
E9B0 0D 82 A4 01 46 2F 07 80 : 30
E9B8 A6 13 4B 5D 1D 51 3F 11 : 1F
E9C0 51 3F 0F 51 3F 2D 5C 3F : F7
E9C8 01 80 BD 15 82 BD 03 4C : E1
E9D0 85 05 82 BC 11 51 34 17 : 75
E9D8 82 BC 07 6E 8D 03 82 BE : 83
E9E0 1B 48 35 03 68 B5 17 53 : 22
E9E8 37 13 46 85 01 66 6D 23 : 0C
E9F0 80 AE 07 82 AD 03 51 34 : EC
E9F8 05 51 34 13 67 94 13 5C : 07
---------------------------------
SUM: 61 9E CE FC 97 A6 16 9F 8A0F

EA00 36 0F 82 97 0F 45 5E 2B : 3B
EA08 80 97 05 50 2D 03 82 9D : BB
EA10 07 82 A6 0D C0 0B 0B 28 : 3A
EA18 32 32 0F 23 00 00 03 82 : 1B
EA20 22 06 82 25 05 62 44 B1 : 2B
EA28 01 24 43 63 B1 04 63 B2 : 95
EA30 03 43 82 62 00 FF 33 60 : BC
EA38 00 01 09 01 09 0B 0E 0B : 38
EA40 0E 02 21 02 21 0E 31 0E : A1
EA48 31 0C 2A 0C 2A 02 35 02 : D6
EA50 35 04 37 04 37 02 54 02 : 03
EA58 54 07 60 07 60 02 8A 02 : B0
EA60 8A 04 8E 04 8E 02 9E 02 : 50
EA68 A6 09 B0 09 AC 02 C1 02 : D9
EA70 C1 05 C3 05 C3 02 D5 02 : 2A
EA78 D5 07 D7 07 D7 02 EA 02 : 7F
---------------------------------
SUM: A3 FA 46 34 71 DF 38 5C 303D

EA80 EA 06 EC 06 EC 02 F7 02 : C9
EA88 F7 0C F9 0C F9 02 FF 02 : 04
EA90 F8 C9 FF FF 01 18 09 18 : F9
EA98 09 11 0E 11 0E 17 22 17 : 97
EAA0 22 14 24 14 24 17 42 17 : 02
EAA8 42 10 4E 10 4E 15 5B 15 : 83
EAB0 5B 17 8B 17 8B 14 8E 14 : 55
EAB8 8E 17 9E 17 A6 10 B1 10 : D1
EAC0 AE 17 CB 17 CB 14 CE 14 : 68
EAC8 CE 17 DF 17 DF 11 E2 11 : BE
EAD0 E2 17 F7 17 F7 12 F9 12 : 1B
EAD8 F9 17 FF 17 FA E0 FF FF : FE
EAE0 65 0E 75 0C 75 06 7B 06 : F0
EAE8 7B 08 77 08 77 12 7B 12 : 18
EAF0 7B 14 75 14 75 11 65 0E : 11
EAF8 73 D6 11 50 4D 03 50 4D : 97
---------------------------------
SUM: 54 9A 9F 48 E0 C6 50 2C BEA5

EB00 01 81 2F 01 80 CF 05 45 : 4B
EB08 7D 01 46 8D 15 82 C4 03 : AF
EB10 83 34 03 82 C4 19 83 34 : D0
EB18 03 51 55 03 83 33 0D 4C : BB
EB20 86 09 83 33 03 82 C4 07 : 95
EB28 66 AD 09 82 C3 17 82 6A : 64
EB30 01 50 93 07 82 6B 09 80 : 61
EB38 6F 05 83 33 07 83 34 0B : F3
EB40 4D 7D 11 51 4E 13 82 C4 : D3
EB48 01 53 4E 13 80 8E 21 4B : 2F
EB50 55 01 81 37 01 82 B3 07 : 4B
EB58 80 B7 05 83 33 1F 6F D6 : 56
EB60 05 82 C4 07 53 64 03 80 : 8C
EB68 C7 07 83 34 0B 81 36 0B : 52
EB70 48 75 33 40 C5 01 40 35 : 6B
EB78 0D 82 9B 01 83 63 03 82 : 96
---------------------------------
SUM: A4 1A 69 9C D3 AF 1D F2 3C63

EB80 9B 01 83 63 19 82 C4 07 : E8
EB88 83 35 05 51 4E 19 4B 8D : 4D
EB90 01 6B 65 0F 82 C5 01 83 : AB
EB98 34 0B 53 55 25 81 37 01 : C5
EBA0 48 4D 03 82 C4 01 83 34 : 96
EBA8 05 4B 8D 39 83 34 07 82 : 56
EBB0 C4 2B 50 63 13 83 35 07 : 74
EBB8 4B 96 07 83 35 0D 5C 5F : 68
EBC0 09 82 C5 0F 47 4C 03 82 : 77
EBC8 C5 0B 6D B6 07 83 35 0F : C1
EBD0 83 35 01 53 5C 15 82 C5 : C4
EBD8 09 49 5D 09 82 C5 07 51 : 57
EBE0 45 0F 83 35 07 6C 86 07 : 0C
EBE8 83 35 03 50 53 03 66 BE : 85
EBF0 15 83 35 03 4B 86 01 83 : 25
EBF8 35 0F 82 C5 01 83 35 03 : 47
---------------------------------
SUM: 1B E6 F4 27 6F C7 45 26 5CAA

EC00 82 C5 0F C0 12 0F 28 00 : 5F
EC08 19 19 14 01 08 07 82 52 : 2A
EC10 0D 82 56 07 23 02 59 06 : 70
EC18 22 42 C5 0A 42 54 C6 27 : B6
EC20 42 55 C2 0A B2 61 42 B2 : 6A
EC28 08 B3 82 63 B1 00 FF    : 50
---------------------------------
SUM: 14 AA 82 3F E2 CD 0A 31 7300
```


▶「X-BASICポケットリファレンスブック」を見て「わ〜！　こんなのがほしかったんだ」と涙の感激をしてしまいました。大きさもちょうどいいし，私のように記憶の足りない奴には大変便利な付録です。さっそくディスプレイと本体の間にはさんでおきました。
前野　千絵(22)東京都

リスト3 MAIN

```
4E00 01 03 1A 3E 0C ED 79 01 : CF
4E08 00 18 21 64 5C 16 00 ED : FC
4E10 51 03 7E ED 79 23 0B 14 : 7A
4E18 7A FE 0E 20 F2 11 00 30 : D9
4E20 21 00 E0 D5 C5 CD 35 4E : EB
4E28 CD 60 4E C1 D1 1C AF B3 : 8B
4E30 20 F1 C3 95 4E E5 D5 01 : 72
4E38 D0 1F AF ED 79 01 D0 3F : 14
4E40 3E 00 CD 59 4E 01 D0 2F : B2
4E48 D1 D5 3E 20 CD 59 4E 01 : 79
4E50 D0 37 D1 7B CD 59 4E E1 : A8
4E58 C9 ED 79 03 15 20 FA C9 : 2A
4E60 06 16 0E 00 16 17 1E 18 : 8D
4E68 3E 08 08 D9 F3 01 01 1A : 36
4E70 ED 78 F2 70 4E ED 78 FA : 74
4E78 75 4E D9 08 ED A3 42 ED : 63
-----------------------------------
SUM: F8 69 9D 0F 71 81 4C 66 CC39

4E80 A3 43 ED A3 06 16 08 3E : D8
4E88 0B 3D C2 89 4E 08 0C 3D : 32
4E90 C2 7C 4E FB C9 CD 52 51 : C0
4E98 CD B7 5B 21 02 5D 11 03 : 73
4EA0 5D 01 71 01 36 00 ED B0 : A3
4EA8 3E FF 32 5D 61 21 00 06 : 54
4EB0 22 70 5E CD 3A 5C CD CC : EC
4EB8 51 CD A6 5A 06 C8 C5 21 : D2
4EC0 80 E1 36 20 11 81 E1 01 : 2B
4EC8 60 09 ED B0 CD 4E 5A C1 : 3C
4ED0 C5 3E 47 90 38 12 FE 11 : 33
4ED8 38 02 3E 11 21 68 68 11 : 8B
4EE0 0A 0E 01 28 09 CD 7D 51 : E5
4EE8 C1 C5 3E 6F 90 38 0C 21 : 28
4EF0 88 66 11 08 3C 01 12 0A : 60
4EF8 CD 7D 51 C1 C5 78 FE 14 : AB
-----------------------------------
SUM: 48 D0 48 9E C7 54 30 E6 EAE6

4F00 30 0C 01 22 17 11 01 1C : A4
4F08 21 BE 6E CD 17 5A CD 9C : F4
4F10 5B C1 10 AA 21 DA 6E 01 : 40
4F18 92 37 3E 17 CD F1 53 CD : FC
4F20 A6 51 FE 43 28 12 CD C3 : 02
4F28 51 EE FF E6 60 20 04 06 : AE
4F30 01 18 8B 3E 01 32 59 61 : CF
4F38 CD CC 51 AF 32 FF 5C 21 : 47
4F40 FF 05 22 00 5D 3E 20 32 : 13
4F48 06 5D 21 00 0F 22 FD 5C : 0E
4F50 CD A6 5A CD B7 5B 01 00 : AD
4F58 30 21 F1 6E 3E 0F CD F1 : BB
4F60 53 01 42 30 3E 07 CD F1 : C9
4F68 53 ED 5F 47 CD C8 5B 10 : E6
4F70 FB CD 52 51 3A 59 61 3D : 9C
4F78 87 87 87 87 C6 80 67 2E : F7
-----------------------------------
SUM: 2D 50 9E 50 43 0B F0 BC DA34

4F80 00 7E 32 5D 61 2E 04 22 : C2
4F88 74 5E 2E 00 3E 06 84 67 : 2F
4F90 22 78 5E 24 24 24 E5 11 : 5A
4F98 3B 61 01 13 00 ED B0 E1 : 2E
4FA0 24 22 50 61 21 AA AA 22 : 8E
4FA8 76 5E 21 FF FF 22 3C 61 : B2
4FB0 CD 23 51 CD 3F 51 CD 4F : BA
4FB8 57 3A 01 5D FE FF CA DA : 90
4FC0 50 3A 37 61 F6 00 28 E8 : 28
4FC8 06 19 78 C6 0F 32 39 61 : 38
4FD0 3E 41 90 32 38 61 0E 08 : F0
4FD8 3A 01 5D FE FF CA DA 50 : 89
4FE0 C5 CD 23 51 CD 86 58 CD : 7E
4FE8 3F 51 C1 0D 20 EA 10 DA : 52
4FF0 3E 02 32 37 61 3E FF 32 : 79
4FF8 5D 61 21 80 F1 36 20 11 : B7
-----------------------------------
SUM: FC A8 55 8A 9B A2 6A B2 910A

5000 81 F1 01 60 09 ED B0 CD : 46
5008 23 51 CD 86 58 CD BB 53 : FA
5010 CD 3F 51 CD BB 53 3A 01 : 73
5018 5D FE FF CA DA 50 3A 37 : BF
5020 61 FE 02 28 E2 06 46 3A : F1
5028 01 5D FE FF CA DA 50 C5 : 14
5030 CD 23 51 CD 86 58 CD C8 : 81
5038 5B 3E 80 BC D4 0A 51 CD : D1
5040 3F 51 C1 10 E2 06 3C C5 : 4A
5048 CD 23 51 C1 C5 78 FE 1E : 5B
5050 D4 0A 51 CD 3F 51 C1 10 : 5D
5058 EE 21 59 61 34 7E FE 07 : 80
5060 C2 69 4F 3D 32 59 61 3E : E1
5068 02 32 37 61 06 02 11 CC : B1
5070 F3 21 2F 6F 0E 24 C5 ED : 96
5078 A0 13 D5 E5 06 04 C5 CD : 09
-----------------------------------
SUM: 7D A9 35 1E 62 6F 88 AA C576

5080 23 51 CD 3F 51 C1 10 F6 : 98
5088 E1 D1 C1 0D 20 E8 EB D5 : 48
5090 11 F8 01 19 D1 EB 10 DC : CB
5098 06 C8 C5 CD 23 51 C1 C5 : 5A
50A0 21 FD 5C 35 23 23 23 36 : 4E
50A8 A0 78 FE A0 30 23 36 FF : 3E
50B0 3E A0 90 F5 21 F6 6A 11 : F5
50B8 07 38 01 14 07 CD 7D 51 : F6
50C0 F1 D6 28 38 0C 21 7E 6C : 3E
50C8 11 07 40 01 10 10 CD 7D : C3
50D0 51 CD 3F 51 C1 10 C3 C3 : 05
50D8 95 4E 06 78 C5 3E FF 32 : 95
50E0 00 5D CD 23 51 3A 37 61 : 70
50E8 F6 00 C4 86 58 CD 97 55 : 51
50F0 C1 C5 3E 78 90 21 37 69 : 8D
50F8 11 07 40 01 12 0C CD 7D : C1
-----------------------------------
SUM: D1 50 FB 34 CD A1 EB 7D 9A2C

5100 51 CD 9C 5B C1 10 D5 C3 : 7E
5108 95 4E CD C8 5B EB 2A 38 : 20
5110 61 7A E6 07 D6 03 84 67 : 8C
5118 7B E6 1F D6 10 85 6F CD : 27
5120 D5 55 C9 CD A6 51 CD B2 : 36
5128 53 CD 3B 5B CD 46 5A CD : F0
5130 DD 51 CD 54 54 CD 17 5C : E3
5138 2A 70 5E CD 4C 51 C9 CD : F8
5140 FE 55 CD 97 55 CD 17 5C : 4C
5148 CD 9C 5B C9 2B 7C B5 20 : 09
5150 FB C9 21 00 F0 36 20 11 : 3C
5158 01 F0 01 00 0C ED B0 21 : BC
5160 00 E0 36 20 11 01 E0 01 : 29
5168 00 0C ED B0 21 72 5E 36 : D0
5170 00 11 73 5E 01 C5 02 ED : 97
5178 B0 C9 C3 95 4E FE AA D0 : 97
-----------------------------------
SUM: 68 CE 40 6C 12 DA 7F 79 5049

5180 FE 22 30 0E E5 C5 D5 CB : A8
5188 3F C6 C8 47 CD 96 51 D1 : 99
5190 C1 E1 CD 17 5A C9 D5 7E : FC
5198 FE 21 38 01 70 23 15 20 : 20
51A0 F6 D1 1D 20 F1 C9 C5 01 : 84
51A8 00 19 3E E6 ED 79 ED 78 : 08
51B0 ED 78 FE 1B CC B9 51 C1 : 15
51B8 C9 CD C3 51 EE FF E6 60 : DD
51C0 C0 18 F6 01 0E 1C ED 49 : 2F
51C8 05 ED 78 C9 01 00 30 11 : 75
51D0 D0 07 26 20 ED 61 03 1B : 89
51D8 7A B3 20 F8 C9 3A 01 5D : A6
51E0 3D FE 80 38 01 AF 01 4C : F0
51E8 30 CD 0B 54 01 0C 30 CD : 66
51F0 24 54 3A 00 5D FE 0A 38 : 4F
51F8 2C D6 04 32 00 5D FE B0 : 43
-----------------------------------
SUM: 74 CD 96 7F 38 0E 53 A7 6C7A

5200 D0 FE 80 DA 2A 52 3A 01 : DF
5208 5D F6 00 20 05 3D 32 01 : E8
5210 5D C9 FE FF C8 21 FD 5C : 65
5218 34 34 34 4E 23 46 21 77 : EB
5220 5C CD 14 5A C9 CD B1 52 : 30
5228 38 71 CD C3 51 2A FD 5C : 0D
5230 E5 CD 7F 53 22 FD 5C E1 : E0
5238 F5 3A 00 5D FE 0A 30 0C : D0
5240 E5 CD FE 52 E1 30 05 22 : 3A
5248 FD 5C 30 00 F1 F5 3A 00 : A9
5250 5D E6 08 20 0C ED 4B FD : AC
5258 5C 21 77 5C F5 CD 14 5A : 80
5260 F1 F1 1F 1F F5 30 06 AF : FA
5268 32 8F 5E 18 0C 3A 8F 5E : 6A
5270 F6 00 CC 20 53 3D 32 8F : 33
5278 5E F1 1F 30 06 AF 32 90 : 15
-----------------------------------
SUM: 3E D7 27 69 81 29 5B 15 95FA

5280 5E 18 18 3A 90 5E F6 00 : AC
5288 20 11 3A FF 5C 3C FE 05 : 05
5290 20 01 AF 32 FF 5C 3E 01 : 9C
5298 32 90 5E 3A FF 5C 87 87 : C3
52A0 87 21 07 6F 06 00 4F 09 : 7C
52A8 01 32 30 3E 08 CD F1 53 : BA
52B0 C9 ED 5B FD 5C DD 21 83 : EB
52B8 60 06 0F C5 AF DD BE 00 : 84
52C0 28 19 DD 7E 04 92 FE 02 : 32
52C8 30 11 DD 7E 02 93 FE 06 : 35
52D0 30 09 DD 36 00 00 EB C1 : F8
52D8 C3 EA 52 01 09 00 DD 09 : EF
52E0 C1 10 D8 CD FE 52 D0 2A : C0
52E8 FD 5C 3E FF 32 00 5D CD : F2
52F0 D5 55 21 00 0F 22 FD 5C : D5
52F8 21 01 5D 35 3F C9 2A FD : E3
-----------------------------------
SUM: 80 DF 7D 48 90 3B F0 8E 8234

5300 5C CD 27 5B 23 3A 5D 61 : C6
5308 0E 02 06 04 7E FE 21 38 : EF
5310 03 FE 50 D8 23 10 F5 11 : 62
5318 5C 00 19 0D 20 EC AF C9 : 06
5320 3A FF 5C 87 21 8B 5C 06 : 2A
5328 00 4F 09 46 23 4E C5 AF : 83
5330 4F 06 06 21 4D 60 11 09 : 43
5338 00 BE 20 01 0C 19 10 F9 : 0D
5340 79 C1 B8 3E 02 D8 C5 11 : E0
5348 7B 5E 79 87 87 87 81 4F : B7
5350 06 00 21 95 5C 09 01 09 : 2B
5358 00 ED B0 ED 5B FD 5C 21 : 5F
5360 05 01 19 EB 21 7D 5E 7B : 81
5368 86 77 23 23 7A 86 77 D9 : 93
5370 21 4D 60 06 06 CD 38 57 : 36
5378 C1 0C 10 CA 3E 02 C9 47 : F7
-----------------------------------
SUM: B9 BC CF 58 A0 BD DD A6 1989

5380 CB 18 38 07 25 7C FE 04 : C5
5388 30 01 24 CB 18 38 07 24 : 9B
5390 7C FE 1B 38 01 25 CB 18 : D6
5398 38 09 2D 2D 7D FE 08 30 : 4E
53A0 02 2C 2C CB 18 38 09 2C : AA
53A8 2C 7D FE 54 38 02 2D 2D : 8F
53B0 78 C9 21 4D 60 06 06 CD : E8
53B8 03 56 C9 DD 21 4D 60 06 : D3
53C0 06 11 09 00 AF DD B6 00 : 62
53C8 28 1C DD 6E 02 DD 66 04 : D8
53D0 CD 27 5B 7E FE 21 38 0E : 32
53D8 FE 5A 38 0F 23 7E FE 21 : 5F
53E0 38 04 FE 5A 38 05 DD 19 : C7
53E8 10 DA C9 DD 36 00 00 18 : DE
53F0 F5 56 ED 51 23 F5 78 D6 : EF
53F8 10 47 16 87 ED 51 03 ED : 22
-----------------------------------
SUM: 9E 11 FB 8A DC 08 1E C3 B775

5400 51 3E 10 80 47 F1 03 3D : 97
5408 20 E7 C9 F5 CB 3F CB 3F : D9
5410 CB 3F CB 3F C6 30 ED 79 : 70
5418 03 03 F1 E6 0F C6 30 ED : CF
5420 79 03 03 C9 16 04 21 02 : 85
5428 5D 7E 23 CD 0B 54 15 20 : 5F
5430 F8 C9 21 05 5D 06 04 B7 : 05
5438 8E 27 77 2B 3E 00 10 F8 : 9D
5440 3A 06 5D 47 3A 04 5D 90 : 0F
5448 D8 21 01 5D 34 78 C6 30 : F9
5450 32 06 5D C9 DD 21 91 5E : 4B
5458 06 19 C5 DD 7E 00 F6 00 : 35
5460 CA 41 55 DD 7E 01 F6 00 : B2
5468 28 07 3D DD 77 01 C3 41 : C5
5470 55 DD 7E 06 F6 00 CA 52 : C8
5478 55 3D DD 77 06 DD 6E 02 : 39
-----------------------------------
SUM: 81 80 C0 E1 5D 00 D0 66 F7FB

5480 DD 7E 04 85 FE B4 D2 4C : B4
5488 55 FE 06 DA 4C 55 6F DD : 20
5490 77 02 DD 66 03 DD 7E 05 : 1F
5498 84 FE 3C D2 4C 55 FE 04 : 33
54A0 DA 4C 55 67 DD 77 03 CB : 04
54A8 3C CB 3D 44 4D DD 6E 0B : 2B
54B0 DD 66 0C CD 14 5A FD 21 : A8
54B8 4D 60 DD 5E 02 DD 56 03 : 20
54C0 CB 3A CB 3B 06 06 C5 D5 : B1
54C8 FD 7E 00 F6 00 28 34 FD : CA
54D0 6E 02 FD 66 04 CD 8D 55 : 86
54D8 30 29 42 4B C5 21 00 65 : 31
54E0 CD 14 5A CD 34 5C 3E 01 : D7
54E8 CD 32 54 E1 FD 36 00 00 : 67
54F0 AF DD BE 00 28 0D DD 35 : 91
54F8 00 20 08 CD D5 55 3E 09 : 66
-----------------------------------
SUM: 1C 7F 1C CA D6 D6 60 F7 58D3

5500 CD 32 54 01 09 00 FD 09 : 63
5508 D1 C1 10 BA D5 1C 1C 14 : 7D
5510 DD 7E 0D DD 86 0E DD 77 : 2D
5518 0E 30 0B EB CD 83 56 ED : C7
5520 5F CB 3F DD 77 0E D1 3A : D6
5528 00 5D FE 0A 30 13 2A FD : CF
5530 5C 7C 92 3C FE 03 30 09 : E0
5538 7D 93 C6 04 FE 08 DC EA : A6
5540 52 11 0F 00 DD 19 C1 05 : 2E
5548 C2 5A 54 C9 DD 36 00 00 : 4C
5550 18 EF DD 6E 07 DD 66 08 : A4
5558 7E DD 77 04 23 7E DD 77 : CB
5560 05 23 7E DD 77 06 23 7E : A1
5568 87 87 47 3A 59 61 CB 3F : 53
5570 80 DD 77 0D 23 54 5D 23 : D8
5578 23 7E F6 00 20 06 DD 5E : F8
-----------------------------------
SUM: 9A 14 FA 09 CB 44 7F 6D D758

5580 09 DD 56 0A DD 73 07 DD : 7A
5588 72 08 C3 71 54 7C 92 FE : 0E
5590 02 D0 7D 93 FE 08 C9 DD : 8E
5598 21 FD 5F 06 0A C5 DD 7E : AD
55A0 00 F6 00 28 27 4F DD 46 : B7
55A8 01 DD 6E 02 DD 66 03 E5 : 79
55B0 11 06 0A CD 17 5A E1 11 : 51
55B8 3C 00 19 DD 75 02 DD 74 : FA
55C0 03 11 7E 66 ED 52 38 04 : 73
55C8 DD 36 00 00 11 04 00 DD : 05
55D0 19 C1 10 C9 C9 E5 CD 2E : 5C
55D8 5C 21 FD 5F 11 04 00 06 : F4
55E0 0A 3E 00 BE 28 05 19 10 : 5C
55E8 FA E1 C9 EB E1 01 02 02 : 75
55F0 ED 42 EB 73 23 72 23 01 : 46
55F8 20 65 71 23 70 C9 21 4D : C0
-----------------------------------
SUM: 52 7A 36 B5 3D 4D 41 5B 1BE2

5600 60 06 15 C5 7E F6 00 28 : DC
5608 6A E5 11 7B 5E 01 09 00 : 43
```

▶ 定価600円，なにかワクワクするのは僕だけだろうか。　　石川　英行(18)愛知県

```
5610 ED B0 2A 7C 5E ED 5B 80 : 69
5618 5E 19 7C FE 5A 30 5E FE : D7
5620 05 38 5A 22 7C 5E 4F 2A : 0C
5628 7E 5E ED 5B 82 5E 19 7C : 99
5630 FE 1E 30 49 FE 04 38 45 : 14
5638 22 7E 5E 6F 26 00 29 29 : E5
5640 29 29 29 54 5D 29 19 06 : 74
5648 F0 09 3A 5D 61 BE 28 2D : 04
5650 23 BE 28 29 2B 3E F0 84 : 0F
5658 67 11 83 5C 3A 7B 5E 87 : F1
5660 CD 3E 58 EB ED A0 ED A0 : 68
5668 D1 D5 21 7B 5E 01 09 00 : AA
5670 ED B0 E1 11 09 00 19 C1 : 72
5678 05 C2 03 56 C9 AF 32 7B : 45
--------------------------------
SUM: EB 6C 0C F2 F6 C4 5B D4 686E

5680 5E 18 E5 3A 00 5D FE 14 : 04
5688 D0 3E 02 32 7B 5E 1E 00 : 39
5690 55 ED 53 7C 5E 54 ED 53 : 03
5698 7E 5E 3A FD 5C C6 03 95 : CD
56A0 CD 0B 5C CB 3F 6F 58 3A : 3F
56A8 FE 5C 3C 94 CD 0B 5C 67 : C5
56B0 50 B5 C8 D5 11 00 01 7C : 30
56B8 BD 38 11 ED 53 82 5E 4C : 72
56C0 65 2E 00 CD F3 5B 29 22 : F9
56C8 80 5E 18 10 11 00 02 ED : 06
56D0 53 80 5E 4D 2E 00 CD F3 : 6C
56D8 5B 22 82 5E D1 2A 80 5E : 36
56E0 29 0E 03 CD F3 5B 7B FE : CE
56E8 01 C4 01 5C 22 80 5E 2A : 4C
56F0 82 5E 29 0E 03 CD F3 5B : 35
56F8 7A FE 01 FE 01 C4 01 5C : 99
--------------------------------
SUM: 92 51 0B C3 C1 C2 64 A4 6352

5700 22 82 5E CD 32 57 C9 ED : 0E
5708 5B FD 5C C5 D5 E5 CD C8 : C8
5710 5B 7C E6 1F D6 0E 83 32 : 75
5718 FD 5C 7D E6 0F D6 08 82 : 2B
5720 32 FE 5C E1 E5 CD 83 56 : F8
5728 E1 D1 C1 10 DE ED 53 FD : 9E
5730 5C C9 D9 21 83 60 06 0F : 17
5738 11 09 00 AF BE 28 05 19 : CD
5740 10 FA D9 C9 11 7B 5E EB : 81
5748 01 09 00 ED B0 D9 C9 3A : 83
5750 77 5E 07 D8 3A 7A 5E F6 : BC
5758 00 28 05 3D 32 7A 5E C9 : 3D
5760 2A 78 5E 7E 23 FE 01 28 : C8
5768 1B FE 02 CA 45 58 FE 03 : 83
5770 20 05 3E 01 32 37 61 FE : 2C
5778 00 20 E8 7E 32 7A 5E 23 : B3
--------------------------------
SUM: 42 1C 7E EA E9 B1 A3 14 9591

5780 22 78 5E C9 7E 23 E5 6F : B6
5788 26 00 29 29 29 11 80 70 : A2
5790 19 EB 1A 13 CD C8 5B BD : DE
5798 E1 1A 38 EA 13 4E CB 21 : 6A
57A0 0C 23 1A FE 02 20 06 ED : 5C
57A8 5F E6 0F 81 4F 46 23 E5 : 72
57B0 08 AF 08 C5 CD 2C 58 38 : 0D
57B8 39 C1 C5 D5 1A 47 13 EB : F3
57C0 ED A0 EB FE 00 28 30 FE : CC
57C8 01 28 39 FE 02 28 28 FE : B0
57D0 03 28 43 23 23 36 00 23 : 0D
57D8 EB ED A0 ED A0 2B 2B 01 : 5C
57E0 04 00 ED B0 CD C8 5B EB : 7C
57E8 23 72 D1 C1 10 C5 E1 C3 : A0
57F0 63 57 C1 E1 C3 63 57 08 : E1
57F8 77 C6 04 08 23 36 AF 23 : 74
--------------------------------
SUM: CB 62 59 6E 47 FA E4 AB 6F4A

5800 71 23 18 CF 36 00 23 36 : 0A
5808 AF 23 08 F5 81 4F F1 C6 : 56
5810 06 08 71 23 18 BD 08 77 : F6
5818 C6 04 08 23 36 AF 23 E5 : E2
5820 CD C8 5B 7D E6 0F 81 E1 : C4
5828 77 23 18 A7 21 91 5E 06 : 6F
5830 19 AF BE C8 C5 01 0F 00 : 23
5838 09 C1 10 F6 37 C9 83 5F : B2
5840 3E 00 8A 57 C9 5E 23 56 : BF
5848 CB 22 23 7E 23 E5 32 E6 : AE
5850 5C 7A 32 E9 5C 7B CB 3F : D2
5858 87 87 C6 03 32 F3 5C ED : 45
5860 5F 32 F4 5C 7B 87 87 5F : C9
5868 87 83 5F 16 00 21 77 6F : 86
5870 19 22 F1 5C CD 2C 58 38 : 11
5878 09 11 E6 5C EB 01 0F 00 : 57
--------------------------------
SUM: 46 B8 A9 D7 B5 AB 91 0C 5010

5880 ED B0 E1 C3 63 57 3A 37 : 6C
5888 61 FE 02 20 68 3A 49 61 : CD
5890 FE C0 30 06 CD 16 59 CD : FD
5898 3B 59 CD 5F 59 CD 72 59 : B1
58A0 CD 9C 59 CD D6 59 2A 38 : 20
58A8 61 25 7D D6 05 6F DD 21 : 4B
58B0 4D 60 06 06 C5 DD 7E 00 : D9
58B8 F6 00 28 31 FE 06 28 2D : A8
58C0 DD 7E 02 95 FE 06 30 25 : 4B
58C8 DD 7E 04 94 FE 03 30 1D : 41
58D0 E5 DD 6E 02 DD 66 04 CD : 46
58D8 D5 55 21 3B 61 35 7E FE : 98
58E0 FF 20 05 3E 01 32 37 61 : 2D
58E8 DD 36 00 00 E1 11 09 00 : 0E
58F0 DD 19 C1 10 BF 2A 38 61 : 49
58F8 ED 5B 4C 61 43 4A CB 39 : 86
--------------------------------
SUM: 12 E0 8B 37 AD 7A 20 4C C554

5900 CB 38 B7 ED 42 44 4D 2A : A4
5908 50 61 CD 17 5A 3A 00 5D : 86
5910 FE 0A DC B1 52 C9 F5 21 : C6
5918 3E 61 CD 0F 5A 30 51 3A : 90
5920 3C 61 47 3A 38 61 80 32 : 69
5928 38 61 FE 26 38 04 FE 55 : 4C
5930 38 3E 78 ED 44 32 3C 61 : EE
5938 18 36 C9 F5 21 40 61 CD : 9B
5940 0F 5A 30 2C 3A 3D 61 47 : E4
5948 3A 39 61 80 32 39 61 FE : 1E
5950 09 38 04 FE 17 38 19 78 : 23
5958 ED 44 32 3D 61 18 11 F5 : 1F
5960 21 42 61 CD 0F 5A 30 08 : 32
5968 2A 38 61 06 02 CD 07 57 : F6
5970 F1 C9 F5 21 44 61 CD 0F : 51
5978 5A 30 F5 21 DD 5C 11 7B : 65
--------------------------------
SUM: F0 BC 26 02 33 F8 AF 32 AD3A

5980 5E 01 09 00 ED B0 2A 38 : 67
5988 61 ED 5F E6 07 D6 03 84 : F7
5990 32 7F 5E 7D 32 7D 5E CD : 66
5998 32 57 F1 C9 F5 21 48 61 : 02
59A0 CD 0F 5A 30 16 2A 38 61 : 3F
59A8 25 25 25 22 4A 61 CD C8 : D1
59B0 5B 7C CB 3F CB 3F CB 3F : F5
59B8 32 49 61 2A 4A 61 7D FE : 2C
59C0 06 38 AD D6 05 6F 22 4A : A1
59C8 61 44 4D 11 07 07 21 F4 : 26
59D0 68 CD 17 5A 18 9A F5 3A : 87
59D8 46 61 F6 00 CA 70 59 06 : 36
59E0 08 11 F5 5C C5 2A 38 61 : F2
59E8 1A 3C 3C E6 7F 12 13 D5 : F1
59F0 11 3E 6E CD 3E 58 1A 85 : BF
59F8 4F 13 1A 84 47 21 25 69 : F6
--------------------------------
SUM: 39 05 22 BB 47 84 3B F2 8B93

5A00 11 03 06 0D 05 CD 17 5A : 6A
5A08 D1 C1 10 D8 C3 70 59 7E : 84
5A10 23 86 77 C9 11 02 06 78 : 7A
5A18 FE 1E D0 79 FE 5A D0 E5 : 72
5A20 60 69 42 4B CD 27 5B D1 : 76
5A28 C5 E5 1A 13 FE 21 38 09 : 37
5A30 08 3A 5D 61 BE 28 02 08 : F0
5A38 77 23 10 EE E1 01 60 00 : DA
5A40 09 C1 0D 20 E3 C9 3A 59 : 36
5A48 61 FE 06 D2 CD 5A DD 21 : 5C
5A50 08 5D 0E 03 06 14 C5 DD : 32
5A58 6E 00 DD 66 01 DD 56 02 : E7
5A60 3E 04 91 F5 82 FE 08 DD : 2D
5A68 77 02 38 14 D6 08 DD 77 : F7
5A70 02 2D 20 0C ED 5F E6 0F : 9C
5A78 C6 5A 6F ED 5F E6 1F 67 : 47
--------------------------------
SUM: 04 BC 7C 31 9C 69 57 3A 0A0B

5A80 DD 75 00 DD 74 01 CD 27 : 98
5A88 5B 3E 20 BE 30 03 F1 18 : B3
5A90 09 F1 3D 87 87 87 DD 86 : 2F
5A98 02 77 11 03 00 DD 19 C1 : 44
5AA0 10 B4 0D 20 AF C9 DD 21 : 67
5AA8 08 5D 06 3C C5 CD C8 5B : 5C
5AB0 7C E6 1F DD 77 01 7D E6 : 39
5AB8 7F 3C DD 77 00 ED 5F E6 : 41
5AC0 07 DD 77 02 01 03 00 DD : 3E
5AC8 09 C1 10 E0 C9 DD 21 08 : 89
5AD0 5D 06 3C DD 7E 00 F6 00 : F0
5AD8 28 35 DD 34 02 DD 34 02 : 83
5AE0 DD 7E 02 F5 E6 03 4F FE : 88
5AE8 02 30 03 DD 35 00 F1 1F : 57
5AF0 1F 1F E6 04 81 C6 18 4F : D6
5AF8 DD 6E 00 DD 66 01 CD 27 : 83
--------------------------------
SUM: C6 62 08 7B 62 73 A5 48 5561

5B00 5B 3E 20 BE 38 01 71 11 : 32
5B08 03 00 DD 19 10 C5 C9 CD : 64
5B10 C8 5B 7C FE 02 30 F0 ED : AC
5B18 5F E6 1F DD 77 01 DD 36 : CC
5B20 00 5A DD 75 02 18 E0 C5 : 6B
5B28 7D 6C 26 00 29 29 29 29 : B3
5B30 29 44 4D 29 09 4F 06 E0 : 21
5B38 09 C1 C9 21 77 5E CB 06 : 5A
5B40 30 4E 3A 37 61 FE 02 28 : 78
5B48 47 21 88 F1 11 87 F1 01 : 6B
5B50 F6 08 ED B0 ED 5B 74 5E : B5
5B58 F6 00 28 03 11 56 61 21 : 0A
5B60 76 5E CB 06 30 03 1B 1B : 0E
5B68 1B 21 D8 F1 3A 5D 61 32 : 2F
5B70 7F 5B 0E 03 1A 13 D5 06 : F3
5B78 08 16 20 1F 30 02 16 00 : A5
--------------------------------
SUM: AF B1 59 65 90 90 10 D0 AF3F

5B80 72 11 60 00 19 05 20 F1 : 12
5B88 D1 0D 20 E8 ED 53 74 5E : F8
5B90 21 88 F1 11 88 E1 01 F6 : 0B
5B98 08 ED B0 C9 01 50 30 21 : 10
5BA0 88 E1 1E 18 16 50 04 ED : F6
5BA8 A3 03 15 20 F9 D5 11 10 : CA
5BB0 00 19 D1 1D 20 EE C9 01 : DF
5BB8 50 20 26 27 11 80 07 ED : 42
5BC0 61 03 1B 7A B3 20 F8 C9 : 8D
5BC8 F5 C5 D5 ED 5B 72 5C ED : 92
5BD0 4B 75 5C CD E2 5B 22 72 : BA
5BD8 5C ED 5F 32 74 5C D1 C1 : 3C
5BE0 F1 C9 21 00 00 3E 10 29 : 52
5BE8 CB 23 CB 12 30 01 09 3D : 42
5BF0 20 F5 C9 AF 06 10 29 17 : E3
5BF8 2C 91 30 02 2D 81 10 F6 : A3
--------------------------------
SUM: EC 4C DB 67 96 35 43 AD D74B

5C00 C9 7C EE FF 67 7D EE FF : 03
5C08 6F 23 C9 FE 81 30 03 06 : 13
5C10 01 C9 ED 44 06 FF C9 0E : D7
5C18 08 11 5A 61 1A F6 00 28 : 0C
5C20 05 3D 12 CD 4D 5C 0C 13 : E9
5C28 79 FE 0B 20 EF C9 21 5B : D6
5C30 61 36 10 C9 21 5A 61 36 : 82
5C38 0D C9 0E 00 21 55 5C 7E : 34
5C40 23 3C C8 3D 28 03 CD 4D : A9
5C48 5C 0C 18 F3 C9 06 1C ED : 4B
5C50 49 05 ED 79 C9 32 00 00 : AF
5C58 08 00 00 19 28 00 00 00 : 49
5C60 00 00 00 FF 6F 50 59 38 : 4F
5C68 1F 02 19 1C 00 07 00 00 : 5D
5C70 00 00 33 E9 00 83 03 5B : FD
5C78 5C 5D 5E 5F 60 7F 80 81 : 56
--------------------------------
SUM: 78 5F B0 7D 37 0A 69 AB 4361

5C80 82 83 84 20 20 A3 A4 A5 : B5
5C88 A6 A7 A8 01 01 02 00 02 : FB
5C90 04 03 01 02 06 01 00 00 : 11
5C98 00 FF 00 03 00 00 01 00 : 03
5CA0 00 00 00 00 03 00 00 01 : 04
5CA8 00 00 00 00 00 02 A0 00 : A2
5CB0 01 00 00 00 00 00 02 60 : 63
5CB8 FF 01 00 FC 00 FF 00 00 : FB
5CC0 00 FF 01 00 FC 00 00 00 : FC
5CC8 00 00 01 01 00 00 00 00 : 02
5CD0 00 03 00 00 01 00 F7 00 : FB
5CD8 00 00 FD 00 00 03 00 00 : 00
5CE0 00 00 00 FC 00 00 00 00 : FC
5CE8 B3 00 FF 00 FF 00 00 00 : B1
5CF0 00 00 00 01 00 00 10 20 : 31
5CF8 30 40 50 60 70 00 00 00 : 90
--------------------------------
SUM: 0F 6F 7B 80 96 AA 4E 28 20DA
```

リスト4　IPL起動用

```
10 CLEAR &HCFFF
20 FOR I=0 TO 30:MEM$(I*255+&HD000,255)=STRING$(255,CHR$(0)):NEX
T
30 LOADM"MAIN   ",&HD000:REC=32 :SIZ=&H17FF:"PUTDISK"
40 LOADM"DATA   ",&HD000:REC=55 :SIZ=&H1AFF:"PUTDISK"
50 LOADM"PCGDATA",&HD000:REC=178:SIZ=&H17FF:"PUTDISK"
60 REC=82:FOR I=1 TO 6
70 LOADM"ROUND"+HEX$(I),&HD000:SIZ=&H1000:"PUTDISK"
80 NEXT
90 A$=   HEXCHR$("01446566 65617432 20202020 20205379")
100 A$=A$+HEXCHR$("730000AA 004E004E 00000000 00002000")
110 DEVO$"1:",0,A$+STRING$(96,CHR$(0))  ,B$
120 END
130 LABEL"PUTDISK"
140 ADR=&HD000
150 A$=MEM$(ADR,128):B$=MEM$(ADR+128,128)
160 DEVO$"1:",REC,A$,B$
170 REC=REC+1:ADR=ADR+256:SIZ=SIZ-256
180 IF SIZ>0 THEN 150
190 RETURN
```

第109部　Small-Cの移植ライブラリ編

●Cコンパイラ完成

アセンブラ，リンカ，ライブラリアン，そしてCコンパイラの移植と進んできたCコンパイラシリーズですが，とうとう完成する日がやってきました。

Cコンパイラはユーザーの作成したプログラムを読み込んでアセンブラのソースファイルを作成しますが，このアセンブラのソースファイルには，printfなどの関数は入っていません。これらの関数はライブラリとして供給されているのです。このライブラリから必要な関数を抽出し，それをまとめてリンクすることによって初めて実行可能なプログラムができあがります。今月お届けするのはこの最後の部分，必要な関数の入ったライブラリの作成です。

市販のCコンパイラを買ったのでは，printfやscanfなどの関数をまとめてライブラリを作成するなどという作業はおそらく経験することはないでしょう。これらの関数のソースファイルを眺めることすら稀なのではないかと思います。今回の作業は，自分でCのライブラリを作り上げていく過程を体験できる，まさに手作りのCコンパイラといった趣です。ぜひ皆さんの手でCコンパイラを完成させてみてください。幸い多くの関数はCP/M版Small-Cのものを流用できますので，作業量はそれほど多くはありません。Cコンパイラの完成まであと一息。頑張りましょう。

●C言語プログラム大募集のお知らせ

多くの作業の末，ようやくたどりついたS-OS初のCコンパイラです。いまやS-OSの標準コンパイラとなった感もあるSLANG，そしてその実数版であるREALとともに，これからも皆で末永く可愛がっていきたいもの。そのためにも，皆さんの熱意あふれる投稿を募集します。ゲームでもシステムプログラムでもなんでも結構です。

今回作成したライブラリはCコンパイラ用の標準的なライブラリですので，S-OSに密着した，アセンブラ代わりのCプログラミングというのは難しいところでしょう。アクションゲームなどのスピードを要求される処理や，より低レベルのプログラムを作成するには，S-OSのファンクションコールを行う関数を集めたライブラリがほしいところです。皆さんのプログラムをお待ちしています。また，こんな関数があったらいいなという提言もお待ちしています。

●S-OSの系譜(23)

どんどんと増殖の手を伸ばし，実働している8ビットマシンのなかでもっとも使用されているシステムとまでいわれたS-OSですが，本家のMZ/X1シリーズでまだその恩恵を受けていないマシンが存在していました。X1turboです。もちろんX1用のS-OS“SWORD”はあったのですが，turboの機能を活かしたS-OSはまだ存在していなかったのです。1987年10月号では，ついに待望のturbo版S-OS“SWORD”が登場しました。

turbo版S-OS“SWORD”の特長をひと言でいうなら，BASICの環境をそのまま使えるS-OSということになります。ユーザーにとってなにより嬉しかったのは，漢字を扱えるようになった点でしょう。コード入力，1字変換だけでなく，音訓変換辞書やシステム/ユーザー辞書も使えるようになっていました。漢字変換は画面最下行を使用して行いますが，ゲームの中には縦25行必要なものがあります。こういったアプリケーションは漢字モードを抜けることで対応するようになっています。このためにCコマンドが追加され，BASICでいうKMODE 1とKMODE 0を切り替えることができるようになっていました。また，BASICで扱えるデバイスがEMMなどを含めすべてサポートされているというのも嬉しいところです。プリンタもBASICで扱えるプリンタのすべてを使用可能。「変身セット」で提言されたRUN&SUBMITルーチンをも内蔵してしまったturbo版S-OSは，S-OSマシンのなかで最高峰の動作環境を持つに至りました。

同じ10月号には，思考型ゲームtiny CORE WARSが収録されています。メモリの中でプログラムとプログラムが殺し合うCORE WARSの魅力は，いかに強いプログラムを作ることができるかというプログラミングテクニックの勝負でもあります。強いプログラムを集めてトーナメントをという案もあったのですが，残念ながら今日までトーナメントは開けずじまいできてしまいました。

そして，待ち望まれていたMAGIC対応のFuzzyBASICコンパイラが登場したのもこの10月号です。作成したマシン語ファイルからサブルーチンのエントリアドレスや変数のアドレスを探し出して表示するクロスリファレンサも用意され，FuzzyBASICで作ったプログラムをマシン語サブルーチンとして使うのが容易になりました。

THE SENTINEL

全機種共通
S-OS"SWORD"要

ライブラリの移植
Small-C

Ishigami Tatsuya
石上 達也

6月号に続いてSmall-Cの移植を行います。今回はS-OSで実際に使用する際に必要になるライブラリの移植を行います。CP/M版から転送したファイルの修正が中心になります。

ライブラリの移植

6月号でのコンパイラ本体の移植が完了したら，続いてライブラリの移植に取りかかります。SWORD用のディスケット1枚にライブラリファイルを収めるのは少しばかり窮屈なので，4枚に分けてファイル転送を行います。表1にその区分を示しておきます。その際，ファイルネームは以下のように小文字にし，拡張子'.C'はそのまま，'.MAC'は'.ASM'にしておいてください。ただし，例外的にRDRTL.MACは大文字のままRDRTL.ASMとしてください(ソースのほうを入力した方は，それをそのままSWORDにコンバートすれば結構です)。

例) DOLDDR.MAC → doldr.ASM
　　ISALPHA.C → isalpha.C

なお，SWORD用に変更のあったライブラリはSWORD用のものを使ってください。CP/MからSWORDへの転送が終わったら，各ディスケットに以下の2つのファイルを入れておいてください。

　stdio.h (変更ずみのもの)
　clib.def (新規作成)

以上の作業が終了したら，いよいよコンパイルです。移植したばかりのコンパイラの出番です。コンパイルオプションは，すべて－M －A －P －O －Iでコンパイルしてください。バッチファイルで作業を行うと，どこかにコンパイルエラーがあったりした場合，パソコンの前に誰かいないと，エラーを抱えたまま作業が続行されてしまいますので，1ステップずつ各自の手で確認しながら入力したほうがよいと思います。具体的なコンパイラの使用方法は前の「コンパイラの使い方」を見てください。

ライブラリのコンパイルが終了したら，次はアセンブル，アセンブルが完了したら，リンクです。表2にバッチファイル形式にまとめておきましたので，このとおりに行います。このとき，リンクする順番とか決まっているので，必ず指定どおりに行ってください。

以上の作業で，clib.LIBというファイルができました。このファイルは，以後Small-Cのライブラリファイルとして利用していくことになります。

Cを実際に使う

では，実際に，Small-Cの使い方を修得するために簡単なプログラムをコンパイルしてみます（といっても，ここまでくるのに，ライブラリのコンパイルでさんざん使っているので，覚えてしまった方も多いでしょうが)。

なんといっても，C言語の第1歩といっ

表1：ライブラリファイルの分け方

（*印　は新規作成）	+free.C	strcat.C
（+印　は変更があったファイル）	*poll.ASM	strchr.C
以下，ライブラリファイル1に収録	abs.C	strcmp.C
*RDRTL.MAC	atoi.C	strcpy.C
*libid.C	atoib.C	strlen.C
*ungetc.C	dtoi.C	strncat.C
*fflush.C	isalnum.C	strncmp.C
rewind.C	isalpha.C	strncpy.C
*ctell.C	isascii.C	以下，ライブラリファイル4に収録
*ctellc.C	isatty.C	strrchr.C
*cseek.ASM	iscntrl.C	toascii.C
fread.C	isdigit.C	tolower.C
+read.C	isgraph.C	toupper.C
fwrite.C	islower.C	utoi.C
+write.C	isprint.C	xtoi.C
+grabuf.C	ispunct.C	pad.C
frebuf.C	isspace.C	+feof.C
*printf.C	isupper.C	+ferror.C
+fprintf.C	isxdigit.C	+clearerr.C
*scanf.C	以下，ライブラリファイル3に収録	+delay.ASM
+fscanf.C	itoa.C	+dolddr.ASM
prntf.C	itoab.C	+doldir.ASM
*putlist.ASM	itod.C	+inp.ASM
calloc.C	itoo.C	+outp.ASM
以下，ライブラリファイル2に収録	itou.C	+topofmem.ASM
malloc.C	itox.C	*delete.ASM
Ualloc.C	left.C	*unlink.ASM
avail.C	lexcmp.C	+rename.C
*cfree.ASM	otoi.C	min.C
	reverse.C	max.C
	sign.C	*CALL.ASM

たらこれしかありません。K&Rの7ページ，ANSI規格準拠の第2版でもやっぱり7ページの「最初のCプログラム」より"Hello World"と表示するプログラム

```
main( ) {
  printf("Hello World¥n") ; }
```

これを，なんらかのエディタを用いて入力します。1990年11月号の山田君のEDC-Tとか，ずっと昔のE-MATEなどを使います（個人的にはREDAのエディタがいちばん気にいっている）。中括弧の使えないシステムでは6月号の囲みを見て対応してください。ここでは，仮にこのファイル名をHELLO.Cとしておきましょう。

次にSWORDのコマンドラインから，

　# CC : HELLO －M －A －P －O

と入力します。ここで，コンパイラ本体をコンパイルしたときとパラメータの与え方が違うじゃないか，と気づいた方は鋭いですね。6月号での変更がここで効いてきています。

ここで，ディレクトリを表示すると，新たにHELLO.ASMというファイルが作成されているはずです。このファイルがHELLO.Cがコンパイルされたアセンブルリストのファイルです。次にこのファイルをアセンブルします。このアセンブルには本誌1990年7月号のWZDを用います。

　# WZD : =HELLO

これで，HELLO.RELというリロケータブルファイルができています。このファイルとシステムライブラリのファイル（後述のclib.LIB）のリンクを行います。これには，本誌1990年8月号のWLKを用います。

　# WLK : HELLO, CLIB/S, HELLO/N : P

以上でコンパイルは終了です。ディスク上にHELLO.OBJというファイルが作成されていますので，それを実行させてみましょう。

　# HELLO.OBJ :

どうです？　ちゃんと画面上に，

　Hello World

と表示されたでしょう。画面に文字列を表示させるのに結構長い道程でしたが，SWORDの拡張を行っている方はバッチ処理ができますね。

最後に

これで，SWORD上でもC言語が動くようになったわけですが，「史上最強のマシン語モニタ」にはまだまだ貧弱です（Pascalコンパイラがほしいじゃありませんか？）。

昔，私が「MAKEがほしいよー」なんていっていたら，このコーナーのある常連さんから投稿がありました。こういうのはたいへん嬉しいですね。近いうちにこのコーナーで発表されると思うので，皆さん楽しみに待っていてください。では，またお会いしましょう。

参考文献

1）DDJツールブック　DDJ編集部編　阿部尚子訳　工学社

2）Small-C V2.7 for CP/M　F.A.Scacchitti　CUG library Disk No.222 & 223

表2　ライブラリのリンク

ライブラリファイル1

WLB:＊RDRTL,libid,ungetc＊fflush,rewind,ctell＊ctellc,cseek,fread＊read,fwrite,write＊grabuf,frebuf,printf＊fprintf,scanf,fscanf＊prntf,putlist,calloc＊clib1/r

ライブラリファイル2

WLB:＊malloc＊ualloc,avail,cfree＊free,poll,abs＊atoi,atoib,dtoi＊isalnum,isalpha,isascii＊isatty,iscntrl,isdigit＊isgraph,islower,isprint＊ispunct,isspace,isupper＊isxdigit＊clib2/r

ライブラリファイル3

WLB:＊itoa,itoab,itod＊itoo,itou,itox＊left,lexcmp,otoi＊reverse,sign,strcat＊strchr,strcmp,strcpy＊strlen,strncat,strncmp＊strncpy＊clib3/r

ライブラリファイル4

WLB:＊strrchr,toascii,tolower＊toupper,utoi,xtoi＊pad,feof,ferror＊clearerr,delay,dolddr＊doldir,inp,outp＊topofmem,delete,unlink＊rename,min,max＊clib4/r

ここで，新しいディスケット，ライブラリファイル5を用意して，そこにライブラリファイル1から4までに作成されたclib1.LIB,clib2.LIB,clib3.LIB,clib4.LIBを，転送します。

そして，

　WLB:＊clib1＊clib2＊clib3＊clib4＊clib/r

ここで作成されたclib.LIBというファイルが，これから使うライブラリファイルです。

ただし，機種によってはS-OSのコマンドラインにこれだけの文字数が使えないものもあります。その場合は，それぞれ頭から順番にまとめていって最終的にclib. LIBというライブラリを作成してください。

WZDシリーズについて

WZDシリーズについて，読者の方から，何件か質問/BUGの発見/提案などがあったので，この場を借りて紹介させていただきます。

○WZDで疑似命令WSEGが使えない。

すみません。私のポカでした。リファレンスマニュアル中ではWork Segmentの略でWSEGとなっていたのですが，直前になってMacro-80との互換性を考え「common」という命令に変えていたのをすっかり忘れていました。

○WLBで入力ファイル名と出力ファイル名が同じだと，同じ名前のファイルができてしまう。

ほかのOS上のライブラリアンでは，こういうことは禁止されているものもあるようですがWLBでは，あえて可能としています。このことは，以下のようなときに役立つと思います。

ライブラリファイルfile1.LIBにリロケータブルファイルfile2.RELを追加する。入出力ファイル名を同一にできない場合，

```
#WLB:＊file1＊file2＊temp/R
#K file1.LIB
#N temp file1.LIB
```

としなければなりませんが，WLBでは，

```
#WLB＊file1＊file2＊file/R
```

とすればいいようになっています。

○commonセグメントでPCのあと戻りができない。

以下のような訂正をお願いします。

```
3843 23 44
4423 CD 9A 3E EB CD 19 3E 3A
442B 53 45 FE 04 20 04 22 5A
4433 45 C9 B7 ED 52 DA 3F 3F
443B C3 31 3A
```

以上，京都市の勇崎晶宏さんからの質問＆デバッグ情報でした（勇崎さんはデバッグまでして編集部にお便りをくれたのでした。感謝）。

○データセグメントのデータが1バイトずれる。

WLKに対し以下の訂正を行ってください。

```
3E64 FF 8F
```

以上，和歌山県の中川学さんからのデバック情報でした（どうもありがとうございます）。

○WZDのX68000上での使用について

Small-Cも含めて，WZDシリーズは，ディスクファイルをアクセスする際に，一部SWORDのDOSモジュールを不正使用しています。具体的にいうと，ファイルをオープンした際，そのファイルのディレクトリ情報は，どこのレコードのどの場所に書かれているかということをDOSモジュール内のワーク（$27E1_H$と$27DF_H$）を直接アクセスして，調べています。よって，DOSモジュールの仕事をエミュレータを通して，実行している機種では動作しません。ただいま，対策を検討中です。

システムがでっかいので，ひょっとしたら（というか，ある程度の確率で），バグが潜んでいるかもしれません。WZDシリーズのバグを見つけたら，ぜひ編集部までご連絡ください（私は編集部に常駐しているわけではないので，質問電話に出ることはありません。なるべく文書でお願いします）。

リスト

```
================  stdio.h  ===================
  1: /*
  2: ** STDIO.H -- Standard Small-C Defs
  3: **
  4: ** 3/24/86 -- F. A. Scacchitti
  5: */
  6:
  7: /*
  8: ** I/O paths
  9: */
 10: #define stdin    0
 11: #define stdout   1
 12: #define stderr   2
 13:
 14: /*
 15: ** special codes
 16: */
 17: #define ERR    (-2)
 18: #define EOF    (-1)
 19:
 20: /*
 21: ** logical states
 22: */
 23: #define YES       1
```

```
24: #define NO      0
25: #define TRUE    1
26: #define FALSE   0
27:
28: /*
29: ** ascii characters
30: */
31: #define NULL    0
32: #define CR     13
33: #define BELL    7
34: #define SPACE  ' '
35: #define NEWLINE CR
36:
37: /*
38: ** special types
39: */
40: #define FILE int

===============  clib.def  =================
 1: #define FLSIZE 0x12 /* File Size. */
 2: #define FLDTADR 0x14 /* Start Address. */
 3: #define FLEXADR 0x16 /* Exec Address. */
 4: #define FLPNT 0x32 /* The FILE Pointor. */
 5: #define NEXTP 0x39 /* offset to next-character pointer in I/O structure */
 6: #define UNUSED 0x3B /* offset to unused-positions-count in I/O structure */
 7: #define ERRFLG 0x3C /* offset to Error-flg in I/O structure */
 8: #define UNGOT 0x3D /* offset to char ungotten by ungetc() */
 9: #define FLAG 0x3E /* file-type flag byte (in unused part of FCB) */
10: #define FCBSIZE 0x3F /* size, in bytes, of an FCB */

===============  libid.C  =================
 1: #include <stdio.h>
 2:
 3: libid() {
 4:
 5: puts("¥n¥n This library created 7/8/86 by F. A. Scacchitti¥n");
 6:
 7: }

===============  ungetc.C  =================
 1:
 2: /*
 3: ** ungetc.c by fas 8/30/84
 4: */
 5:
 6: #include <stdio.h>
 7: #include <clib.def>
 8:
 9: ungetc(c,fd) char c, *fd;{
10:
11:  if(fd[UNGOT] != 0 ) return(EOF);
12:
13:  fd[UNGOT] = c;
14:  return(c);
15: }

===============  fflush.C  =================
 1:
 2: /*
 3: ** fflush.c by F. A. Scacchitti 9/15/84
 4: */
 5:
 6: #include <stdio.h>
 7: #include <clib.def>
 8:
 9: fflush(fd) char*fd; {
10:
11:     if(fd[FLAG] & WRTFLG) {
12:  if(fd[UNUSED] && fd >= 0x100) {
13: #asm
14:  POP BC
15:  POP DE
16:  PUSH DE
17:  PUSH BC
18:  CALL _WRITE##
19:  LD HL,0
20:  RET NC
21:  DEC HL
22:  RET
23: #endasm
24:  }
25:    }
26:    return(NULL);
27: }

===============  ctell.C  =================
 1:
 2: /*
 3: ** ctell.c by T.Ishigami        2/08/91
 4: */
 5:
 6: #define NOCCARGC
 7:
 8: #include <stdio.h>
 9: #include <clib.def>
10:
11: ctell(fd) char *fd; {
12:
13: return(fd[FLPNT] * 0x100 + fd[UNUSED]);
14:
15: }

===============  ctellc.C  =================
 1:
 2: /*
 3: ** ctellc.c by T.Ishigami       2/09/91
 4: */
 5:
 6: #define NOCCARGC
 7:
 8: #include <clib.def>
 9:
10: ctellc(fd) char *fd; {
11:
12: return(fd[UNUSED]);
13:
14: }

===============  cseek.ASM  =================
 1: ;
 2: ;  cseek(fd,offset,base) by T.Ishigani  2/09/91
 3: ;
 4: ;  Position fd to the 256-byte record indicated by
 5: ; "offset" relative to the point indicated by "base."
 6: ;
 7: ;      BASE     OFFSET-RELATIVE-TO
 8: ; 0        first record
 9: ; 1        current record
10: ; 2        end of file (last record + 1)
11: ;
12: ; Returns NULL on success, else EOF.
13: ;
14:
15: ; cseek(fd, offset, base) int fd, offset, base; {
16:
17:  INCLUDE clib.ASM
18:
19: cseek::
20:  POP BC
21:  POP IX ;fd
22:  POP DE ;offset
23:  POP HL ;base
24:  PUSH HL
25:  PUSH DE
26:  PUSH IX
27:  PUSH BC
28:
29:  LD A,H
30:  AND A
31:  LD A,L
32:  LD HL,-1
33:  RET NZ
34:
35:  AND A  ;case 0:
36:  LD HL,0  ;   offset += 0;
37:  JP Z,CC1  ;   break;
38:
39:  DEC A
40:  JR NZ,CC2  ;case 1:
41:     ;   offset += ftell(fd);
42:  LD H,(IX+FLPNT) ;   break;
43:  LD L,(IX+UNUSED)
44:  JR CC1
45:     ;case 2:
46: CC2: DEC A  ;   offset += File-Size;
47:  LD HL,-1  ;   break;
48:  RET NZ
49:
50:  LD L,(FLSIZE) ;default: return(EOF);
51:  LD H,(FLSIZE+1) ;   break;
52:     ;   break;
53: CC1: ADD HL,DE
54:
55:  LD A,(IX+FLAG) ;if(fd[FLAG] != WRTFLG) {
56:  AND 1
57:  JR NZ,CC3
58:
59:  PUSH HL  ;if(offset > fd[FLSIZE])
60:  LD E,(IX+FLSIZE+1) ; return(EOF);
61:  LD D,(IX+FLSIZE)
62:  OR A
63:  SBC HL,DE
64:  POP DE
65:  LD HL,-1
66:  RET NC
67:
68:  LD (IX+FLPNT),D
69:  LD (IX+UNUSED),E
70:
71:  PUSH IX
72:  POP HL
73:  LD D,0
74:  ADD HL,DE
75:  LD (IX+NEXTP),L
76:  LD (IX+NEXTP+1),H
77:
78:  PUSH IX
79:  POP DE
80:  CALL _READ##
81:  LD HL,0
82:  RET NC
83:  DEC HL
84:  RET
85:
86: CC3:
87:  PUSH HL  ;write(fd);
88:  PUSH IX
89:  POP DE
90:  CALL _WRITE##
91:  POP DE
92:  LD HL,-1
93:  RET C
94:  EX DE,HL
95:
96:  LD (IX+FLPNT),L
97:  LD (IX+UNUSED),H
98:
99:  PUSH IX
100:  EX (SP),HL
101:  LD D,0
102:  ADD HL,DE
103:  LD (IX+NEXTP),L
104:  LD (IX+NEXTP+1),H
105:  POP HL
106:
107:  LD E,(IX+FLSIZE+1) ;if(offset < fd[FLSIZE])
108:  LD D,(IX+FLSIZE) ;     _read(fd);
109:  OR A
110:  SBC HL,DE
111:
112:  PUSH IX
113:  POP DE
114:  CALL C,_READ##
115:  LD HL,0
116:  RET NC
117:  DEC HL
118:  RET

===============  read.C  =================
 1: /*
 2: ** read.c by F. A. Scacchitti 3/22/86
 3: */
 4:
 5: #include <stdio.h>
 6: #include <clib.def>
 7:
 8: extern int zzbuf;
 9:
10: static int i, n;
11: static char *tbuff, flag;
12:
13: read(fd,buffer,cnt) int fd, cnt; char *buffer; {
14:
15:    tbuff = &zzbuf;
16:    n=0;
17:
18:  while(cnt >0){
19:     if((flag = _sqread(fd)) != NULL) {
20:        *(fd + ERRFLG) = flag;
21:        return(n);
22:     }else
23:        i = 0;
24:     while(i <= 256 && cnt > 0){ buffer[n] = tbuff[i];
25:       i++; cnt--; n++; }
26:  }
27:     return(n);
28: }

===============  write.C  =================
 1: /*
 2: ** write.c by F. A. Scacchitti 11/24/84
 3: */
 4:
 5: #include <stdio.h>
 6: #include <clib.def>
 7:
 8: extern int zzbuf;
 9:
10: static int i, n;
11: static char *tbuff, flag;
12:
13: write(fd,buffer,cnt) int fd, cnt; char *buffer; {
14:
15:    tbuff = &zzbuf;
16:
17:    n=0;
```

▶なぜ，TAKERUは旭川にこないのか。なぜ，イベントは旭川で開かれないのか。なぜ，編集部は旭川スタルヒン球場を知っているのか。なぜだ～！　太田　哲雄(16)北海道

```
 18:  while(cnt >0){
 19:     i = 0;
 20:     while(i <= 256 && cnt-- > 0) tbuff[i++] = buffer[n++];
 21:
 22:     if((flag = _sqwrite(fd)) != NULL) {
 23:         *(fd + ERRFLG) = flag;
 24:         return(n);
 25:     }
 26:  }
 27:  return(n);
 28: }

===============  grabuf.C  =================
  1:
  2: /*
  3: ** grabufc.c by fas 8/30/84
  4: */
  5:       _
  6: #include <stdio.h>
  7: #include <clib.def>
  8:
  9:    static int buf;
 10:
 11: grabuf() {
 12:
 13:    if((buf = grabio()) != NULL) return(buf+FCBSIZE);
 14:    return(NULL);
 15: }

===============  printf.C  =================
  1: #define NOCCARGC
  2: /*
  3: ** Yes, that is correct.  Although these functions use an
  4: ** argument count, they do not call functions which need one.
  5: */
  6: #include stdio.h

===============  fprintf.C  =================
  1: #define NOCCARGC
  2: /*
  3: ** Yes, that is correct.  Although these functions use an
  4: ** argument count, they do not call functions which need one.
  5: */
  6: #include stdio.h
  7:
  8:
  9: /*
 10: ** fprintf(fd, ctlstring, arg, arg, ...) - Formatted print.
 11: ** Operates as described by Kernighan & Ritchie.
 12: ** b, c, d, o, s, u, and x specifications are supported.
 13: ** Note: b (binary) is a non-standard axtension.
 14: */
 15:
 16: static int  arg, left, pad, cc, len, maxchr, width;
 17: static char *ctl, *sptr, str[17];
 18:
 19: fprintf(argc) int argc; {
 20:   int *nxtarg;
 21:   nxtarg = CCARGC() + &argc;
 22:   return(_print(*(--nxtarg), --nxtarg));
 23:   }
 24:
 25: /*
 26: ** _print(fd, ctlstring, arg, arg, ...)
 27: ** Called by fprintf() and printf().
 28: */
 29: _print(fd, nxtarg) int fd, *nxtarg; {
 30:   cc = 0;
 31:   ctl = *nxtarg--;
 32:   while(*ctl) {
 33:     if(*ctl!='%') {fputc(*ctl++, fd); ++cc; continue;}
 34:     else ++ctl;
 35:     if(*ctl=='%') {fputc(*ctl++, fd); ++cc; continue;}
 36:     if(*ctl=='-') {left = 1; ++ctl;} else left = 0;
 37:     if(*ctl=='0') pad = '0'; else pad = ' ';
 38:     if(isdigit(*ctl)) {
 39:       width = atoi(ctl++);
 40:       while(isdigit(*ctl)) ++ctl;
 41:       }
 42:     else width = 0;
 43:     if(*ctl=='.') {
 44:       maxchr = atoi(++ctl);
 45:       while(isdigit(*ctl)) ++ctl;
 46:       }
 47:     else maxchr = 0;
 48:     arg = *nxtarg--;
 49:     sptr = str;
 50:     switch(*ctl++) {
 51:       case 'c': str[0] = arg; str[1] = NULL; break;
 52:       case 's': sptr = arg;          break;
 53:       case 'd': itoa(arg,str);       break;
 54:       case 'b': itoab(arg,str,2);    break;
 55:       case 'o': itoab(arg,str,8);    break;
 56:       case 'u': itoab(arg,str,10);   break;
 57:       case 'x': itoab(arg,str,16);   break;
 58:       default:  return (cc);
 59:       }
 60:     len = strlen(sptr);
 61:     if(maxchr && maxchr<len) len = maxchr;
 62:     if(width>len) width = width - len; else width = 0;
 63:     if(!left) while(width--) {fputc(pad,fd); ++cc;}
 64:     while(len--) {fputc(*sptr++,fd); ++cc; }
 65:     if(left) while(width--) {fputc(pad,fd); ++cc;}
 66:     }
 67:   return(cc);
 68:   }

===============  scanf.C  =================
  1: #define NOCCARGC  /* no argument count passing */
  2: /*
  3: ** Yes, that is correct.  Although these functions use an
  4: ** argument count, they do not call functions which need one.
  5: */
  6: #include stdio.h

===============  fscanf.C  =================
  1: #define NOCCARGC  /* no argument count passing */
  2: /*
  3: ** Yes, that is correct.  Although these functions use an
  4: ** argument count, they do not call functions which need one.
  5: */
  6: #include stdio.h
  7:
  8: static char *carg, *ctl, *unsigned;
  9: static int  *narg, wast, ac, width, ch, cnv, base, ovfl, sign;
 10:
 11: /*
 12: ** fscanf(fd, ctlstring, arg, arg, ...) - Formatted read.
 13: ** Operates as described by Kernighan & Ritchie.
 14: ** b, c, d, o, s, u, and x specifications are supported.
 15: ** Note: b (binary) is a non-standard extension.
 16: */
 17: fscanf(argc) int argc; {
 18:   int *nxtarg;
 19:   nxtarg = CCARGC() + &argc;
 20:   return (_scan(*(--nxtarg), --nxtarg));
 21:   }
 22:
 23: /*
 24: **  _scan(fd, ctlstring, arg, arg, ...) - Formatted read.
 25: ** Called by fscanf() and scanf().
 26: */
 27: _scan(fd,nxtarg) int fd, *nxtarg; {
 28:   ac = 0;
 29:   ctl = *nxtarg--;
 30:   while(*ctl) {
 31:     if(isspace(*ctl)) {++ctl; continue;}
 32:     if(*ctl++ != '%') continue;
 33:     if(*ctl == '*') {narg = carg = &wast; ++ctl;}
 34:     else             narg = carg = *nxtarg--;
 35:     ctl += utoi(ctl, &width);
 36:     if(!width) width = 32767;
 37:     if(!(cnv = *ctl++)) break;
 38:     while(isspace(ch = fgetc(fd))) ;
 39:     if(ch == EOF) {if(ac) break; else return(EOF);}
 40:     ungetc(ch,fd);
 41:     switch(cnv) {
 42:       case 'c':
 43:         *carg = fgetc(fd);
 44:         break;
 45:       case 's':
 46:         while(width--) {
 47:           if((*carg = fgetc(fd)) == EOF) break;
 48:           if(isspace(*carg)) break;
 49:           if(carg != &wast) ++carg;
 50:           }
 51:         *carg = 0;
 52:         break;
 53:       default:
 54:         switch(cnv) {
 55:           case 'b': base =  2; sign = 1; ovfl = 32767; break;
 56:           case 'd': base = 10; sign = 0; ovfl =  3275; break;
 57:           case 'o': base =  8; sign = 1; ovfl =  8191; break;
 58:           case 'u': base = 10; sign = 1; ovfl =  6553; break;
 59:           case 'x': base = 16; sign = 1; ovfl =  4095; break;
 60:           default:  return (ac);
 61:           }
 62:         *narg = unsigned = 0;
 63:         while(width-- && !isspace(ch=fgetc(fd)) && ch!=EOF) {
 64:           if(!sign)
 65:             if(ch == '-') {sign = -1; continue;}
 66:             else sign = 1;
 67:           if(ch < '0') return (ac);
 68:           if(ch >= 'a')       ch -= 87;
 69:           else if(ch >= 'A') ch -= 55;
 70:           else                ch -= '0';
 71:           if(ch >= base || unsigned > ovfl) return (ac);
 72:           unsigned = unsigned * base + ch;
 73:           }
 74:         *narg = sign * unsigned;
 75:       }
 76:     ++ac;
 77:     }
 78:   return (ac);
 79:   }

===============  putlist.ASM  =================
  1: ;
  2: ;
  3: ; putlist(c)
  4: ;
  5: ; T. Ishigami 2/6/91
  6: ;
  7: ;
  8: _LPRNT EQU 1FDCH
  9: PUTLIST::
 10:  POP BC
 11:  POP HL
 12:  PUSH HL
 13:  PUSH BC
 14:
 15:  LD A,L
 16:  CALL _LPRNT
 17:
 18:  LD H,0   ; return(c & 0377)
 19:  RET NC
 20:
 21:  LD HL,-1
 22:  RET
 23:
 24:  END

===============  cfree.ASM  =================
  1:  EXT free

===============  free.C  =================
  1: #define NOCCARGC  /* no argument count passing */
  2: extern char *zzmem;
  3: /*
  4: ** free(ptr) - Free previously allocated memory block.
  5: ** Memory must be freed in the reverse order from which
  6: ** it was allocated.
  7: ** ptr    = Value returned by calloc() or malloc().
  8: ** Returns ptr if successful or NULL otherwise.
  9: */
 10: free(ptr) char *ptr; {
 11:    return (zzmem = ptr);
 12:    }

===============  ferror.C  =================
  1: /* ferror.c    by      T.Ishigami        2/09/91
  2: **
  3: ** Returns true only if a file error has occurred.
  4: **
  5: */
  6:
  7: #include <clib.def>
  8:
  9: ferror(fd) char *fd; {
 10:
 11:    return(fd[ERRFLG]);
 12:
 13: }

===============  clearerr.C  =================
  1: /*
  2: ** clearerr.c by T.Ishigami   2/09/91
  3: **
  4: ** Clears errors and end of file marks in fd.
  5: **
  6: */
  7:
  8: #include <stdio.h>
  9: #include <clib.def>
 10:
 11: clearerr(fd) char *fd; {
 12:
 13: fd[ERRFLG] = NULL;
 14:
 15: }

===============  delay.ASM  =================
  1: ;
  2: ; delay(n)
  3: ;  int n;
  4: ;
  5: ; n = number of milliseconds to delay
  6: ;
  7: DELAY::
  8:  POP HL  ; Return address
  9:  POP DE  ; Delay Value
 10:  PUSH DE  ; Restore Stack
 11:  PUSH HL  ;
 12: DELAY1:     ;*   51  (overhead) 12.5 usec.
 13:  LD BC,123  ;*    10
 14:  CALL DELAY2  ;* 3963  (17 + 10 + (123 X 32))
```

▶ 先日,「麻雀放浪記」を放送していた。以前にも何度か観ているけど久し振りに観ると,セリフの裏がよくわかって面白かった。以前は「女郎」の意味さえわからなかったのだ。けれど,ほかの大切なことも忘れてしまったような気もする。　家田　貴之(23)愛媛県

```
 15:  DEC DE  ;#    5
 16:  LD A,D  ;#    5
 17:  OR E  ;#    4
 18:  JP NZ,DELAY1 ;#   10
 19:  RET   ;--------
 20:      ;  3997 cycles @ 250 nanosec per
 21:      ;    -------------------------
 22:      ;   0.99925 milliseconds
 23: ;
 24: ;
 25: ; Delay Loop set for 10 usec. per count based on 4 MHz clock
 26: ;
 27: DELAY2:
 28:  DEC BC  ;#   5
 29:  OR A  ;#   4
 30:  OR A  ;#   4
 31:  LD A,B  ;#   5
 32:  OR C  ;#   4
 33:  JP NZ,DELAY2 ;#  10
 34:  RET    ;#
 35:     ;#------
 36:     ;#  32 cycles @ 250 nanosec per = 12.5 usec.
 37:  END

===============  doldr.ASM  =================
  1: ;
  2: ; lddr(source, dest, n)
  3: ;
  4: DOLDDR::
  5:  INC SP ; Skip over return address
  6:  INC SP
  7:  POP BC ; Load n
  8:  POP DE ; Load destination
  9:  POP HL ; Load source
 10:  LDDR
 11:  PUSH HL ; Restore stack
 12:  PUSH DE
 13:  PUSH BC
 14:  DEC SP
 15:  DEC SP
 16:  RET
 17:
 18:  END

===============  doldir.ASM  =================
  1: ;
  2: ; doldir(source, dest, n)
  3: ;
  4: DOLDIR::
  5:  INC SP ; Skip over return address
  6:  INC SP
  7:  POP BC ; Load n
  8:  POP DE ; Load destination
  9:  POP HL ; Load source
 10:  LDIR
 11:  PUSH HL ; Restore stack
 12:  PUSH DE
 13:  PUSH BC
 14:  DEC SP
 15:  DEC SP
 16:  RET
 17:
 18:  END

===============  inp.ASM  =================
  1: ;
  2: ; inp(port#)  ; Added 2/84 (fas)
  3: ;
  4: INP::
  5:  POP DE ; Skip over return address
  6:  POP BC ; Load port # into C
  7:  IN L,(C)
  8:  PUSH BC ; Restore stack
  9:  PUSH DE
 10:  LD A,L ; Data was returned in L
 11:  RLCA  ; Sign extend HL
 12:  SBC A,A ;
 13:  LD H,A ; That's it
 14:  RET
 15:
 16:  END

===============  outp.ASM  =================
  1: ;
  2: ; outp(port#,data) ; added 2/84 (fas)
  3: ;
  4: OUTP::
  5:  POP DE ; Skip over return address
  6:  POP HL ; Load data in L
  7:  POP BC ; Load port # in C
  8:  OUT (C),L
  9:  PUSH BC ; Restore stack
 10:  PUSH HL
 11:  PUSH DE
 12:  RET
 13:
 14:  END

===============  topofmem.ASM  =================
  1: ;
  2: ;
  3: ; End of memory function
  4: ; Returns top memory location in HL
  5:
  6: _MEMAX EQU 1F6AH
  7:
  8: TOPOFMEM::
  9:  LD HL,(_MEMAX) ; Get top of free-memory
 10:  RET
 11:
 12:  END

===============  unlink.ASM  =================
  1: ;
  2: ;  unlink(name) char *name:
  3: ;    by T.Ishigami 2/09/91
  4: ;
  5: _FILE EQU 1FA3H
  6: _KILL EQU 2015H
  7:
  8: unlink::
  9:  POP BC
 10:  POP DE
 11:  PUSH DE
 12:  PUSH BC
 13:
 14:  LD A,4
 15:  CALL _FILE
 16:  CALL NC,_KILL
 17:  LD HL,0
 18:  RET NC
 19:  DEC HL
 20:  RET

===============  rename.C  =================
  1: #define NOCCARGC  /* no argument count passing */
  2: #include stdio.h
  3: #include clib.def
  4: /*
  5: ** Rename a file.
  6: **  from = address of old filename.
  7: **     to = address of new filename.
  8: **  Returns NULL on success, else ERR.
  9: */
 10: rename(from, to) char *from, *to; {
 11:   char fcb[FCBSIZE];
 12:   char *ptr;
 13:   ptr = fcb;
 14:   if(strlen(from) > NTSIZE) return(ERR);
 15:   if(strlen(to) > NTSIZE) return(ERR);
 16:   while(*from)  *(ptr++) = *(from++);
 17:   *(ptr++) = ':';
 18:   while(*to)    *(ptr++) = *(to++);
 19:   *ptr = 0;
 20:
 21:   if(Sdos(RENAME, fcb) != 255)
 22:     return(NULL);
 23:   return (ERR);
 24:   }

===============  CALL.ASM  =================
  1: ;
  2: ; CALL.REL ｦﾞ ﾄﾘｺﾏﾚﾀｺﾄｦ ｼﾒｽ
  3: ;
  4: CALL::
  5:
  6: ;
  7: ;----- CALL.ASM: small-c arithmetic and logical library
  8: ;     for Z-80 & S-OS "SWORD" version
  9: ;     (Many routines is changed by T.Ishigami )
 10:
 11: CCDCAL EQU 1F81H
 12:  PUBLIC CCDCAL
 13: ;
 14: CCDDGC::
 15:  ADD HL,DE
 16:  JP CCGCHAR
 17: ;
 18: CCDSGC::
 19:  INC HL
 20:  INC HL
 21:  ADD HL,SP
 22: ;
 23: ;fetch a single byte from the address in HL and sign into HL
 24: CCGCHAR::
 25:  LD A,(HL)
 26: ;
 27: ;put the accum into HL and sign extend through H.
 28: CCARGC::
 29: CCSXT::
 30:  LD L,A
 31:  RLCA
 32:  SBC A,A
 33:  LD H,A
 34:  RET
 35: ;ﾅｵｼﾀ
 36: CCDDGI::
 37:  ADD HL,DE
 38:  LD A,(HL)
 39:  INC HL
 40:  LD H,(HL)
 41:  LD L,A
 42:  RET
 43: ;
 44: CCDSGI::
 45:  INC HL
 46:  INC HL
 47:  ADD HL,SP
 48: ;
 49: ;fetch a full 16-bit integer from the address in HL into HL
 50: CCGINT::
 51:  LD A,(HL)
 52:  INC HL
 53:  LD H,(HL)
 54:  LD L,A
 55:  RET
 56: ;ﾅｵｼﾀ
 57: CCDECC::
 58:  INC HL
 59:  INC HL
 60:  ADD HL,SP
 61:  DEC (HL)
 62:  JP CCGCHAR
 63: ;ﾅｵｼﾀ
 64: CCINCC::
 65:  INC HL
 66:  INC HL
 67:  ADD HL,SP
 68:  INC (HL)
 69:  JP CCGCHAR
 70: ;
 71: CDPDPC::
 72:  ADD HL,DE
 73:  POP BC ;ret addr
 74:  POP DE
 75:  PUSH BC
 76:  LD A,L
 77:  LD (DE),A
 78:  RET
 79: ;ﾅｵｼﾀ
 80: CCDECI::
 81:  INC HL
 82:  INC HL
 83:  ADD HL,SP
 84:  LD E,(HL)
 85:  INC HL
 86:  LD D,(HL)
 87:  DEC DE
 88:  LD (HL),D
 89:  DEC HL
 90:  LD (HL),E
 91:  EX DE,HL
 92:  RET
 93:
 94: ;ﾅｵｼﾀ
 95: CCINCI::
 96:  INC HL
 97:  INC HL
 98:  ADD HL,SP
 99:  LD E,(HL)
100:  INC HL
101:  LD D,(HL)
102:  INC DE
103:  LD (HL),D
104:  DEC HL
105:  LD (HL),E
106:  EX DE,HL
107:  RET
108:
109: ;
110: CDPDPI::
111:  ADD HL,DE
112: CCPDPI::
113:  POP BC ;ret addr
114:  POP DE
115:  PUSH BC
116: ;
117: ;store a 16-bit integer in HL at the address in DE
118: CCPINT::
119:  LD A,L
120:  LD (DE),A
121:  INC DE
122:  LD A,H
123:  LD (DE),A
124:  RET
125: ;
```

```
126: ;inclusive "or" HL and DE into HL
127: CCOR::
128:  LD A,L
129:  OR E
130:  LD L,A
131:  LD A,H
132:  OR D
133:  LD H,A
134:  RET
135: ;
136: ;exclusive "or" HL and DE into HL
137: CCXOR::
138:  LD A,L
139:  XOR E
140:  LD L,A
141:  LD A,H
142:  XOR D
143:  LD H,A
144:  RET
145: ;
146: ;"and" HL and DE into HL
147: CCAND::
148:  LD A,L
149:  AND E
150:  LD L,A
151:  LD A,H
152:  AND D
153:  LD H,A
154:  RET
155: ;
156: ;in all the following compare routines, HL is set to 1 if the
157: ; condition is true, otherwise it is set to 0 (zero).
158: ;
159: ;test if HL = DE
160: ;ﾅｵｼﾀ
161: CCEQ::
162:  OR A
163:  SBC HL,DE
164:  LD HL,1
165:  RET Z
166:  DEC L
167:  RET
168: ;
169: ;test if DE != HL
170: ;ﾅｵｼﾀ
171: CCNE::
172:  OR A
173:  SBC HL,DE
174:  LD HL,1
175:  RET NZ
176:  DEC L
177:  RET
178: ;ﾅｵｼﾀ
179: ;test if DE <= HL (signed)
180: CCLE::
181:  CALL CCCMP
182:  RET NC
183:  DEC L
184:  RET
185: ;ﾅｵｼﾀ
186: ;test if DE > HL (signed)
187: CCGT::
188:  CALL CCCMP
189:  RET C
190:  DEC L
191:  RET
192: ;ﾅｵｼﾀ
193: ;test if DE < HL (signed)
194: CCLT::
195:  CALL CCCMP
196:  DEC HL
197:  RET Z
198:  RET C
199:  INC L
200:  RET
201:
202: ;ﾅｵｼﾀ
203: ;test if DE >= HL (signed)
204: CCGE::
205:  CALL CCCMP
206:  RET Z
207:  RET C
208:  DEC L
209:  RET
210: ;
211: ;common routine to perform a signed compare of HL and DE
212: ; this routine performs HL - DE and sets the conditions:
213: ; carry reflects sign of difference (set means HL < DE)
214: ; zero/non-zero set according to equality.
215: CCCMP::
216:  LD A,H  ;invert sign of HL
217:  XOR 80H
218:  LD H,A
219:  LD A,D  ;invert sign of DE
220:  XOR 80H
221:  LD D,A
222:  SBC HL,DE  ;compare HL and DE
223:  LD HL,1  ;preset true cond
224:  RET
225: ;
226: ;test if DE <= HL (unsigned)
227: ;ﾅｵｼﾀ
228: CCULE::
229:  EX DE,HL
230: ;
231: ;test if DE >= HL (unsigned)
232: ; ﾅｵｼﾀ
233: CCUGE::
234:  OR A
235:  SBC HL,DE
236:  LD HL,1
237:  RET C
238:  RET Z
239:  DEC L
240:  RET
241: ;
242: ;test if DE < HL (unsigned)
243: ; ﾅｵｼﾀ
244: CCULT::
245:  EX DE,HL
246: ;
247: ;test if DE > HL (unsigned)
248: ;ﾅｵｼﾀ
249: CCUGT::
250:  OR A
251:  SBC HL,DE
252:  LD HL,1
253:  RET C
254:  DEC L
255:  RET
256: ;
257: ;shift DE arithmetically right by HL and return in HL
258: ;ﾅｵｼﾀ
259: CCASR::
260:  EX DE,HL
261:  DEC E
262:  RET M
263:  SRL H
264:  RR L
265:  JP CCASR+1
266: ;
```

```
267: ;shift DE arithmetically left by HL and return in HL
268: CCASL::
269:  EX DE,HL
270:  DEC E
271:  RET M
272:  ADD HL,HL
273:  JP CCASL+1
274: ;
275: ;subtract HL from DE and return in HL
276: ;ﾅｵｼﾀ
277: CCSUB::
278:  EX DE,HL
279:  OR A
280:  SBC HL,DE
281:  RET
282: ;
283: ;form the two's complement of HL
284: CCNEG::
285:  LD A,H
286:  CPL
287:  LD H,A
288:  LD A,L
289:  CPL
290:  LD L,A
291:  INC HL
292:  RET
293: ;
294: ;form the one's complement of HL
295: CCCOM::
296:  LD A,H
297:  CPL
298:  LD H,A
299:  LD A,L
300:  CPL
301:  LD L,A
302:  RET
303: ;
304: ;multiply DE by HL and return in HL (signed multiply)
305: ;ｵｵｲﾆ ﾅｵｼﾀ
306: CCMULT::
307: MULT: LD B,H
308:  LD C,L
309:  LD HL,0
310: MULT1: SRL B
311:  RR C  ;BC = BC / 2
312:  JP NC,MULT2
313:  ADD HL,DE
314: MULT2: LD A,C
315:  OR B
316:  RET Z
317:  SLA E
318:  RL D  ;DE = DE * 2
319:  JP MULT1
320: ;
321: ;divide DE by HL and return quotient in HL,remainder in DE (signed divide)
322: ;ｵｵｲﾆ ﾅｵｼﾀ
323: CCDIV::
324:  LD A,D
325:  XOR H
326:  PUSH AF
327:  LD A,D
328:  OR A
329:  CALL M,CCDENEG
330:  LD A,H
331:  OR A
332:  CALL M,CCNEG
333:  LD B,H
334:  LD C,L
335:  LD HL,0
336:  LD A,16
337: CCDIV1: SLA E
338:  RL D  ;DE = DE * 2
339:  ADC HL,HL  ;HL = HL * 2 + CY
340:  INC E  ;with CY = 0
341:  SBC HL,BC
342:  JP NC,CCDIV2
343:  DEC E
344:  ADD HL,BC
345: CCDIV2: DEC A
346:  JP NZ,CCDIV1
347: CCDIV3: POP AF
348:  EX DE,HL
349:  RET P
350:  CALL CCNEG
351:
352: ;
353: ;negate the integer in DE (internal routine)
354: CCDENEG:LD A,D
355:  CPL
356:  LD D,A
357:  LD A,E
358:  CPL
359:  LD E,A
360:  INC DE
361:  RET
362: ;
363: ;logical negation
364: ;ﾅｵｼﾀ
365: CCLNEG::
366:  LD A,H
367:  OR L
368:  INC HL
369:  RET Z  ;if HL = 0 then RET with HL = 1
370:  LD HL,0
371:  RET   ;if HL != 0 then RET with HL = 0
372: ;
373: ; execute "switch" statement
374: ;
375: ; hl = switch value
376: ; (sp) -> switch table
377: ;  dw addr1, value1
378: ;  dw addr2, value2
379: ;  ...
380: ;  dw 0
381: ;  jmp default]
382: ;  continuation
383: ;
384: CCSWITCH::
385:  EX DE,HL ;de = switch value
386:  POP HL ;hl -> switch table
387: SWLOOP: LD C,(HL)
388:  INC HL
389:  LD B,(HL) ;bc -> case addr, else 0
390:  INC HL
391:  LD A,B
392:  OR C
393:  JP Z,SWEND ;default or continuation code
394:  LD A,(HL)
395:  INC HL
396:  CP E
397:  LD A,(HL)
398:  INC HL
399:  JP NZ,SWLOOP
400:  CP D
401:  JP NZ,SWLOOP
402:  LD H,B ;case matched
403:  LD L,C
404: SWEND: JP (HL)
405: ;
406: zzbuf:: NOP
407:  END _LINK**
```

▶シャープブランドの「WIZARDRY」シリーズは出ませんかねえ。アクションゲームもいいけどX68000の「WIZARDRY」もいけると思います。シャープさん，目のつけどころを変えてみては？
鈴木　健也(24)静岡県

全機種共通
システムインデックス

＊以上のアプリケーションは、基本システムであるS-OS "MACE" または S-OS"SWORD" がないと動作しませんのでご注意ください。

INDEX

発売速報

サウンドキャンバスSC-55

Nakano Shuichi 中野 修一

パソコンミュージックに最適のサウンドモジュール，SC-55が発売された。GSスタンダードに対応した初めての製品だ。標準音色と充実したコントロールチェンジは高品質データの互換性をも保証するか？

Rolandの新型音源モジュール，サウンドキャンバスSC-55がついに発売された。この音源は超MIDI規格ともいえるGSスタンダードに基づいた初めての製品だ。

GSスタンダード以外にも，ハーフラックサイズのコンパクトさでも16マルチティンバー，U-220などと同じRS-PCM方式の音源を積んだと思われる音色，別製品のシーケンサ，サウンドブラシとシステムを組んで使えるなど話題は多い。

インプリメンテーションチャートを見ると互換性重視のためエクスクルーシブにはほとんど期待できないことからか，コントロールチェンジ関係が非常に充実していることがわかる。

GSスタンダード

謎に包まれていた標準規格「GSスタンダード」の内容がこれでやっと明るみに出た。マニュアルによると，GSスタンダードでは「キャピタル」と呼ばれる標準の音色セットを持ち，機種依存の音色「バリエーション」を複数持てるようになっている。ドラムセットも複数持てるがスタンダードセットが基本になるようだ。

SC-55の場合，キャピタル(バリエーション番号0）のほかにバリエーション番号8，16に一部の楽器音，1～9に特殊効果音が割り当ててある。MT-32の音色セットはバリエーション番号127に相当する。標準音色であるキャピタルの内容はクラシック，ジャズ，ロック，ポピュラーミュージック，民族音楽と，多彩な分野で使用できるようにバラエティに富んでいる。配列もピアノ，クロマチックパーカッション，オルガン，ギター，ベース，ストリングス&オーケストラ，アンサンブル，ブラス，リード，パイプ，シンセリード，シンセパッド，シンセSFX，エスニック，パーカッシブ，SFXといった16分野に8音ずつと整然とした配列をしている。

そのほか，GSスタンダードの標準機能として，

1)　16パート
2)　最大24音以上
3)　コントロールチェンジのバンクセレクトによって複数の音色セットに対応
4)　プログラムチェンジによるドラムセット切り替えが可能
5)　リバーブ，コーラスのエフェクトを備える。パート単位で指定可能

などが挙げられている。細かい部分についてはマニュアルだけではよくわからない。

MT-32対応のゲームを鳴らす

SC-55はMT-32系列と同じ音色配列も内蔵していることがひとつのウリになっている。そこで，これまで発売されているMT-32対応ゲームをSC-55のMT-32互換配列モードで実行してみた。

ソフトハウスオリジナルの音色を使っているものは当然うまくいかない（無論，まったく聞けないわけではないが)。たとえば，スーパーハングオン，ジェミニウィング，闇の血族，グラナダなどがそうだ。

MT-32の内蔵音色だけで作られたもの(初期のMIDI対応作品すべて，最近ではパロディウスだ！，ファランクスなど）は曲によってはかなりちゃんと演奏するし曲によっては聞き苦しいこともある。

個々の音を聞くと音源方式は違うはずだが，まさしくLA音源の音が再現される(サンプリング系のものを除く)。さすがRoland純正といってよい。ただし，音量のバランスが違っており，全体にドラムが大きく前に出てくる。ファランクスで使っているメインの音色などはよく出ているのだが，メタルサイトでメイン音色に選ばれている音はたいてい小さく沈んでおり，バランスがよくない。

結論としては，一応MT-32の音にかなり似た音は出るが，MT-32用に調整された音楽を聞くには多少無理があるといったところだろうか。

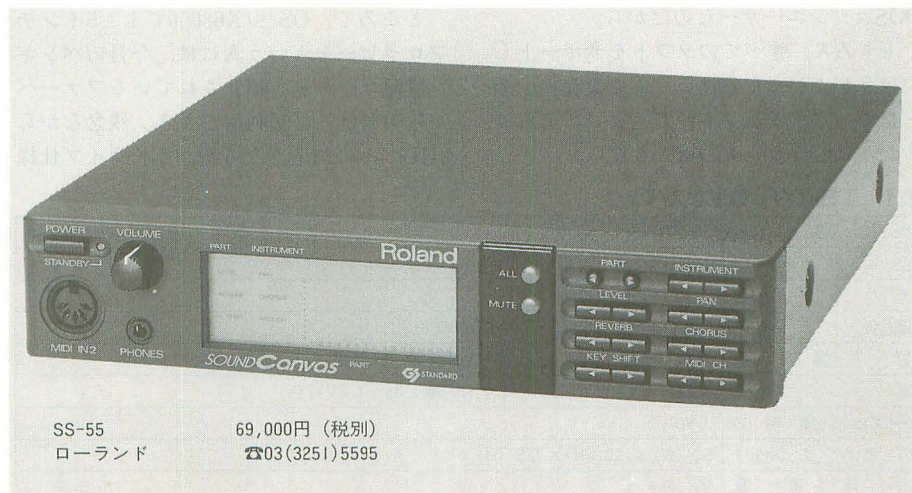
普通のシンセと違い音色変更はそれほど自由ではない。基本音はそのまま，チャンネルごとに設定を変更する。ビブラート，エンベロープ，レゾナンスなどが変更可能。音は変えても楽器は変えないわけだ。

*　　　*　　　*

早くも品薄状態の人気ぶりで，すでにSC-55対応を表明しているゲームも登場している。機能，値段，性能などを考えあわせてみればMT-32に代わる新しい標準機となるのは間違いない。各種ソフトウェアの対応が待たれるところだ。

来月はさらに詳しく紹介したい。

SS-55　　69,000円（税別）
ローランド　☎03(3251)5595

製品紹介

X68000用3.5インチフロッピーディスクドライブ

TS-3XR1

Kaneko Shunichi 金子 俊一

かねてからアナウンスされていたツクモの3.5インチディスクドライブがようやく発売。2HD/2DDの主なフォーマットをサポートするデバイスドライバも付属している。

ツクモ（九十九電機）からX68000用の外付け3.5インチフロッピーディスクドライブが発売された。3.5インチのフロッピーディスクも世間では一般化しており，メディアコンバートで頭を悩ませている人も多かったのではないだろうか。実際に編集部でもメディアコンバータとも呼ばれる専用の32ビットマシンがあるくらいだ。

ノートパソコンやブックパソコンが売れに売れている昨今ではX68000のほかにセカンドマシンとしてそれらの3.5インチパソコンを持っている人も多いと思われる。証拠というほどではないが，実際にこのディスクドライブは人気大爆発のようだ。なにを隠そうこの私もそんなユーザーのひとりなのである。

むふふ相性診断？？％

さっそくレポートしてみたい。まず，気になるX68000との相性を探っていこう。このドライブはHuman68kのCOMMAND.X上で使用することを前提としており，SX-WINDOWやVS，OS-9やCP/M上では動作が保証されていない。つまり，コマンドモードでX68000を動かさない人は使えないということだ。まあ，実害はないだろう。メディア交換の必要性がある人の多くはDOSマシンユーザーなのだから。

もちろん，すべてのソフトをサポートしたほうがよかったのはいうまでもない。あとはオートイジェクトができない。こればっかりはできないものはできない。

私が試してみたところでは，VSでも動作するようだ。ディスクを差し込めばウィンドウはオープンするし，操作も内蔵の5インチドライブに比べてなんら変わりはない。ただし，クローズ/イジェクトをするとディスクのアクセスランプがつきっぱなしになってしまうので注意が必要。このときは，自分でイジェクトしてやればランプは消える。ファイルがオープンしていないのなら，ディスクを抜いてからクローズ/イジェクトするといいようだ。

SX-WINDOWとの相性はやはり悪かった。ドライブの接続がチェックされない。これは正常に動作しないと見てもよさそうだ。X68000の標準フォーマットでもチェックしてくれない。

では，ソフト面で見てみよう。TYPEやCOPYなどのコマンドは問題なく動いた。FORMATやDISKCOPY，DRIVEだって動く。どうやら**Human68k上では２HDの拡張ドライブとして完全に動作する**ようだ。動いて当たり前なのだが，なぜか嬉しい。3.5インチで立ち上がるHuman68kというのもなにか面白いではないか。

＊

以上のことを踏まえたうえで，X68000との相性診断は貴兄にまかせたい。私はよいほうだと思うのだが。

ところで，OS-9/X68000でも3.5インチフロッピーをという人には，今月のペンギン情報コーナーで紹介されているファーベル社のドライブが利用できる。残念ながら2HDのみの対応となるが，２ドライブ仕様でOS-9/X68000用のドライバが付属している。OS-9ユーザーや3.5インチをメインにシステムを組みたい人はそちらを検討されるのもよいだろう。

サポートが恋しい

さて，以上のことは２HDの８セクタフォーマットのときの話であった。ご存じの人も多いとは思うが，同じ２HDでもセクタ長が違うと読み書きは自由に行われないのである。X68000の標準フォーマットは８セクタであり，このほかに15セクタフォーマットというものが存在する。J3100などのタイプで，２HCとも呼ばれる。さらに２DDでは８セクタ，９セクタというフォーマットもある。

ここで２DDという話が出てきたので気がついた人もいるだろう。この3.5インチドライブは２DDも読み書きできるのである。

もともとX68000は拡張用のフロッピーディスクドライブに対しては，２HDは当然として，２DDや２Dですら対応していたのである。２HDの専用ドライブではなかったことに拍手をおくりたい。これならほとんどのDOSマシンの3.5インチディスクはすべて読み書きができるではないか。

これらの２HD８セクタ以外のフォーマットを利用する場合は，デバイスドライバを組み込むことで可能となる。面白いことに内蔵の５インチ２HDドライブを15セクタで扱うこともできる。

組み込み方はちょっとだけあと回しにして，注意点を挙げておこう。X68000ではフォーマットができないのである。これは，FORMAT.Xが２HD15セクタや２DDに対応してないためであるが，この程度はドライブメーカー側で用意しておいてほしかったところだ。いまのところはほかのマシンでフォーマットするしかない。この点に関してはいずれバージョンアップなどでユーザーサポートをしていただけるのではないかと信じて文句はいうまい。

表1　TS-3XR1の主なスペック

項目	仕様
機種名	TS-3XR1
使用ドライブ	JPN DS-33A
ドライブ台数	1台
適合メディア	3.5インチフロッピー　2DD/HD
付属品	接続ケーブル，ゴム足，取扱説明書，保証書
電源	家庭用AC100V（電源ユニット内蔵・85～132V可能）
平均消費電力	6W以下
外形寸法(mm)	57（W）×120（H）×225（D）
本体重量	約1.7kg

増設はカステラのように

X68000の特徴のひとつにデザインが挙げられると思う。少なからず，コンピュータというイメージとはちょっと違った雰囲気を持ったマシンである。まあ“のっぺり”しているといえばそれまでだが。

さらに，こだわる人がユーザーに多いのも特徴といえる。外付けハードディスクやプリンタ，カラーイメージユニットなども同色系を選びたがるようだ。ツクモさんにはぜひともブラックタイプの発売を検討していただきたいところ。オートイジェクト対応だとさらによい。

このドライブは，スペック表を見ればわかるように結構コンパクトである。体積的には長崎屋のハニーカステラといい勝負だろうか。縦置きが基本のようだが，横置きもできる。消費電力が少ないので発熱量も少なく，通風に気をつける必要はない。縦ならちょうどX68000のPROシリーズの高さとほぼ一緒。どうせなら完全に揃っていたほうがよかったかもしれない。

接続はX68000とディスクドライブを結ぶ平行ケーブル1本だけでOK。もちろん，ケーブルは買えばついてくる。あとは電源を入れるだけ。これならメカ音痴の人でも安心して接続できるだろう。

組み込んでみよう

それでは2HCや2DD用のデバイスドライバの組み込み方を解説しよう。

組み込み用のファイルは3つある。

fddhd15.sys (2HD15セクタ用)
fdddd8.sys (2DD8セクタ用)
fdddd9.sys (2DD9セクタ用)

である。2HDの8セクタフォーマットだけはこのようなものがなくても，Human68kの立ち上げ時にドライブの電源が入っていれば自動的に組み込まれる。

組み込みはconfig.sysに，

```
device=fddhd15.sys #0 #1 #2
device=fdddd8.sys #2
device=fdddd9.sys #2
```

の3行を加える。これらはramdisk.sysと同じように登録したドライブごとに順番にドライブ名(A:，B:など)をひとつずつ取っていく。#の次の数字は物理的なドライブナンバーになっているのだが，ハードディスクやRAMディスクの有無は関係ない。#0は内蔵5インチの0ドライブ，#2は3.5インチドライブという意味だ。

上記の場合でハードディスクもRAMディスクもなかったなら，

A：5インチ0ドライブ2HD8セクタ
B：5インチ1ドライブ2HD8セクタ
C：3.5インチドライブ2HD8セクタ
D：5インチ0ドライブ2HC15セクタ
E：5インチ1ドライブ2HC15セクタ
F：3.5インチドライブ2HC15セクタ
G：3.5インチドライブ2DD8セクタ
H：3.5インチドライブ2DD9セクタ

となる。A:からC:までがHuman68kによって自動的に組み込まれ，D:からH:までがデバイスドライバによって組み込まれたドライブである。必要なものだけを組み込むようにしたい。

注意しなければならないことは，物理的に同じドライブに対して，フォーマットが異なれば違うドライブ名を与えているので，入れるディスクによってドライブ名を考えなければならないことだ。たとえば，3.5インチの2HCディスクをドライブに入れた場合，F:ドライブなら正常動作するが，C:，G:，H:の各ドライブでアクセスすると「無効なメディアを使用しました」と表示される。ここらへんはメーカーが苦心したところだと思うが，まだまだ改良の余地があるだろう。某国民機では自動的に判断しているのだから。

ここでは参考のために2DDのデバイスドライバを8セクタ用と9セクタ用の2本を組み込んでいるが，これは避けたほうがいいらしい。マニュアルにもそう書いてある。試しに8セクタフォーマットのフロッピーディスクを9セクタでアクセスしてみたら，アクセスできてしまった。うむ，ちょっとコワイ。おそらく正常には動いていないのだろう。「気づかないで使ったらフォーマットが違っていた」なんて場合では，データを破壊している可能性が高いのである。よって2DDのデバイスドライバは1本組み込むのが正統である。ちなみに2HDではフォーマットが違うとアクセスできないので，心配なく組み込んでほしい。

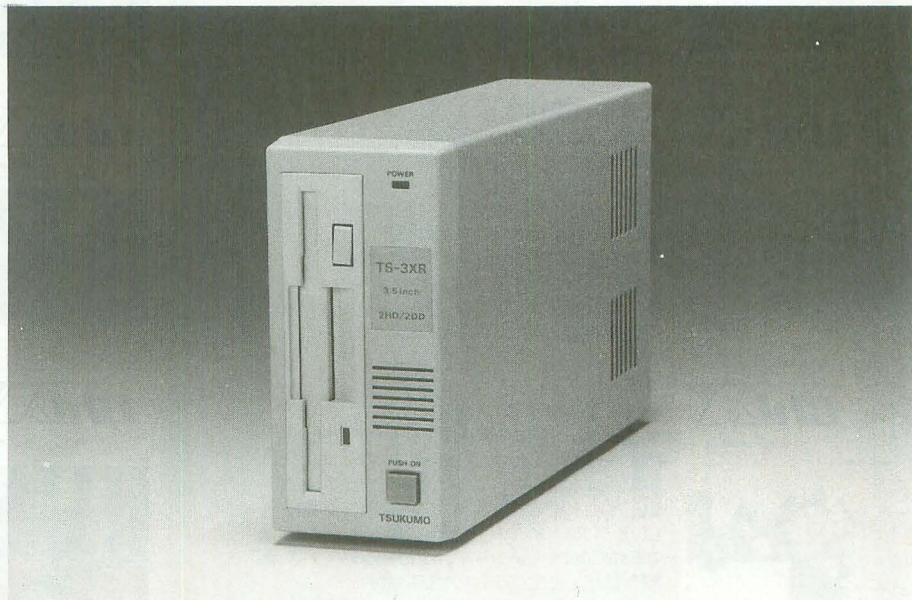
なおDISKCOPYは，転送元と転送先のフォーマットが同じであれば，従来のDISKCOPY.Xが使えた。つまり，2HDの15セクタフォーマット同士や8セクタフォーマット同士なら可能だったわけだ。ちょっと不思議。普通にCOPYでファイルコピーをするならば，フォーマットが違うこと自体はまるで問題にならなかった。たとえば，2DDの9セクタから2HDの8セクタへコピーなどということは可能である。

目のつけどころがツクモでしょ

全体的にみると，2HDなら完全な増設ドライブとして，2HCや2DDならメディアコンバータとして使うのが正解だろう。必要なときだけデバイスドライバを組み込めばいいのだから。

ちなみにこのドライブの価格だが，定価は44,800円のところ，ツクモ特価35,800円(消費税別途1,074円)となっている。自分のところで出しておいてツクモ特価というあたり，結構ノリがいい。

X68000の世界を広げるこの1台，安いとみるか高いとみるか。それはあなた次第である。

3.5インチフロッピーディスクドライブ
TS-3XRI　定価44,800円(税別)
九十九電機　☎03(3251)9911

BACK ISSUES バックナンバー案内

ここには1990年8月号から1991年7月号までをご紹介しました。現在1990年10, 11, 1991年1～7月号の在庫がございます。バックナンバーおよび定期購読の申し込み方法については, 172ページを参照してください。

1990

8月号（品切れ）
特集　ADVANCED 2D GRAPHICS
100号記念特別モニタプレゼント
連載　ショートプロばーてぃ/Z80'Bar/INTEGRAL XI
X-BASIC調理実習/X68000マシン語プログラミング
PurePASCAL/ハードウェア工作入門
●X68000用画像回転プログラム　XROTO.X
LIVE in '90　OMENs OF LOVE/ENDLESS RAIN/ダートフォックス
TCE SOFTOUCH　大航海時代/ウルティマV/プロミストランド
全機種共通システム　リンカWLK

9月号（品切れ）
特集1　日本語を処理するための序章
特集2　ADVANCED 2D GRAPHICS
連載　ショートプロばーてぃ/Z80's Bar/DōGA・CGA
X-BASIC調理実習/マシン語プログラミング
PurePASCAL/ハードウェア工作入門
●清水和人流プログラミング道場
LIVE in '90　風の谷のナウシカ/ラジオ体操第一
THE SOFTOUCH　T&T/D-Again/シムシティー/ギャラガ'88ほか
全機種共通システム　BILLIARDS

10月号
特集　電子音楽術入門
連載　ショートプロばーてぃ/Z80's Bar/DōGA・CGA
マシン語プログラミング/ハードウェア工作入門
清水和人流プログラミング道場
●荻窪圭の大人のためのX68000
●中森章のようこそここへC言語
LIVE in '90　Rise And Fall/PARADOX/キューピー3分クッキング
THE SOFTOUCH　ワールドコースト/ルーンワース/闇の血族/提督の決断
全機種共通システム　ライブラリアンWLB

11月号
特集　理科系のGAME REVIEW
連載　Z80's Bar/DōGA・CGA/カードゲーム
マシン語プログラミング/ハードウェア工作入門
PurePASCAL/X-BASIC調理実習
ようこそここへC言語/INTEGRAL XI
●荻窪圭の大人のためのX68000
LIVE in '90　ピラミッドソーサリアン/ザ・スキーム
THE SOFTOUCH SPECIAL　ラグーン/幻獣鬼/サイバリオン/GUNSHIP他
全機種共通システム　スクリーンエディタEDC-T

12月号（品切れ）
特集　XCのための傾向対策
連載　X-BASICプログラミング調理実習/ハードウェア工作入門
マシン語プログラミング/ショートプロばーてぃ/Z80's Bar
大人のためのX68000/ようこそここへC言語/INTEGRAL XI
●シミュレーションプログラミング入門
●特別企画アナログジョイスティックの制作
LIVE in '90　グラディウスIII/メタルサイト
THE SOFTOUCH SPECIAL　イメージファイト/ジェミニウイング/NAIOUS他
全機種共通システム　STACKコンパイラ

1991

1月号
特集　急接近! SX-WINDOW
特別付録　謹賀新年PRO-68K（5" 2HD）
連載　ハードウェア工作入門/シミュレーションプログラミング入門
DōGA・CGA/ショートプロばーてぃ/大人のためのX68000
PurePASCAL/清水和人流プログラミング道場/X-BASIC調理実習
LIVE in '91　めぞん一刻/涙で綴るパパへの手紙
THE SOFTOUCH　ソル・フィース/銀英伝II/続ダンジョン・マスター他
製作紹介　光磁気ディスクCZ-6MOI
全機種共通システム　ブロックアクションゲームCOLUMNS

2月号
特集1　グラフィックの"実験的"手法
特集2　SX-WINDOWプログラミング
連載　ハードウェア工作入門/シミュレーションプログラミング入門
マシン語プログラミング/大人のためのX68000/Z80's Bar
ショートプロばーてぃ/INTEGRAL XI/ようこそここへC言語
●1900年度 GAME OF THE YEARノミネート発表
LIVE in '91　Misty Blue/スプーンおばさん
THE SOFTOUCH　栄冠は君に/KLAX/ダイナマイト・デューク他
全機種共通システム　ダイスゲームKISMET

3月号
特集　MIDI & MUSIC PROCESSING
連載　ハードウェア工作入門/シミュレーションプログラミング入門
マシン語プログラミング/大人のためのX68000/Z80's Bar
ショートプロばーてぃ/DōGA・CGA/C言語/PurePASCAL
●SXLIFE完結編/ウィンドウシステム大比較
●周辺機器新製品紹介
LIVE in '91　戦いの兜/LITTLE WING/リゾ・ラバ/花
THE SOFTOUCH　アトミック・ロボキッド/スペースズローグ他
全機種共通システム　アクションゲームMUD BALLIN'

4月号
特集　人とゲームのインタフェイス
連載　DōGA・CGA/シミュレーションプログラミング入門
ハードウェア工作入門/ようこそここへC言語/Z80's Bar
ショートプロばーてぃ/清水和人流プログラミング道場
●新連載　吾輩はX68000である/よいこのSX-WINDOW講座
●決定!　1990年度GAME OF THE YEAR
LIVE in '91　Easy Come, Easy Go!/シシリエンヌ
THE SOFTOUCH　メルヘンメイズ/中華大仙/スライス他
全機種共通システム　SLANG用カードゲームDOBON

5月号
特集　新登場!　X68000XVI/XVI-HD
特別付録　黄金週間PRO-68K（5"2HD）
第6回　言わせてくれなくちゃだワ
連載　ハードウェア工作/ようこそここへC言語
大人のためのX68000/X68000マシン語プログラミング
ショートプロばーてぃ/マシンカクテル in Z80's Bar
LIVE in '91　ブービーキッズ/NO.NEW YORK
THE SOFTOUCH　マーブル・マッドネス/シグナトリー/石道他
全機種共通システム　実数型コンパイラ言語REAL

6月号
特集　初心者のための環境構成術
創刊9周年記念Oh!Xアンケート結果大分析大会その1
連載　ハード工作/大人のためのX68000/Z80's Bar/DOGA
ようこそC言語/ショートプロばーてぃ/SX-WINDOW
吾輩はX68000である/マシン語プログラミング
●響子　in CGわーるど
LIVE in '91　暴れん坊将軍/ナディア/POWER HALL他
THE SOFTOUCH　パロディウスだ!/遥かなるオーガスタ/ノスタルジア他
全機種共通システム　S-OS 6周年記念　Small-C　処理系の移植

7月号
特集　Personal Tool, BASIC
別冊付録　X-BASIC　ポケットリファレンスブック
連載　大人のためのX68000/ハード工作/響子 in CGわーるど
ショートプロばーてぃ/SX-WINDOW/吾輩はX68000である
ようこそC言語/Z80's Bar/マシン語プログラミング
●X1用ゲーム　The Master of Payment
LIVE in '91　今すぐKISS ME/歩いていこう
THE SOFTOUCH　パロディウスだ!/ファランクス/スコルピウス/AIII他
全機種共通システム　実数型コンパイラ言語REAL ソースリスト編

愛読者プレゼント

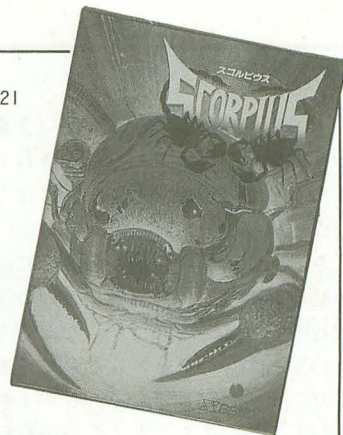
1

新声社　☎03(3293)9321

スコルピウス

X68000用 5"2HD版2枚組　7,800円(税別)

3名

ゲーメストのゲーマー集団が作った，横スクロールタイプのシューティングゲーム。ムズいゲームにはまってみたい方におすすめです。

2

工画堂スタジオ　☎03(3353)7724

サブナック

X68000用 5"2HD版2枚組　7,800円(税別)

3名

石化した妖精を，元の世界に帰してあげるアクションパズルゲーム。といっても，もっぱら使うのはアタマのほうばかり，というくらい難解なゲームだ。

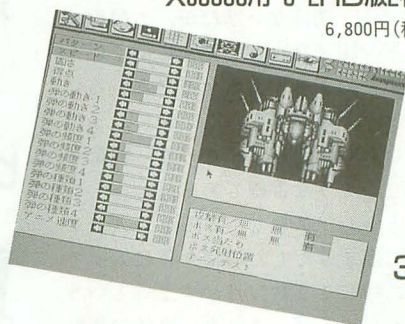
3

ブラザー工業　☎052(824)2493

シューティング68K

X68000用 5"2HD版2枚組

6,800円(税別)

3名

マウスで手軽にシューティングゲームが作れるコンストラクションツール。プログラミングなんてできない，という人にはもってこいです。

4

ファミリーソフト
☎03(3924)5727

TESORITO

10名

ファミリーソフトが作った，非売品CDをプレゼント。ここ1～2年のファミリーソフトのゲームミュージックが17曲収録されています。

プレゼントの応募方法

とじ込みのアンケートはがきの該当項目をすべてご記入のうえ，希望するプレゼント番号をはがき右下のスペースにひとつ記入してお申し込みください。締め切りは1991年8月18日の到着分までとします。当選者の発表は1991年10月号で行います。

5

イマジニア　☎03(3343)8911

MAXISバッジ

10名

シムシティーでお馴染み，アメリカのMAXIS社。その会社で実際に社員バッジとして使用しているものをプレゼント。

6月号プレゼント当選者

1 A栄冠は君に（愛知県）杉森貴之ほか2名　B機甲師団（大阪府）阪本泰博ほか1名　2 Aシムシティー（愛媛県）枝松樹ほか2名　B（三重県）谷口保ほか2名　3 MAGICAL SHOT（東京都）梅津長裕ほか4名　4 Aエメラルドドラゴン（千葉県）桑田義久ほか2名　Bグローディアテレカ（岡山県）鄙伸二ほか2名　5 Aランペルール（東京都）佐藤俊ほか1名　Bハンドブック（東京都）田口博規ほか2名　C CD（神奈川県）野村恵ほか2名　6 パロディウスだ！（北海道）牧野豊ほか2名　7 ルーシーショット（埼玉県）福士三成ほか2名　8 AイースIII（千葉県）御代川由尚ほか2名　Bステッカー（千葉県）吉岡雅幸ほか9名　Cディスクホルダー（東京都）鈴木崇ほか4名　Dタオル（兵庫県）久保田博宣ほか4名　Eマウスマット（岡山県）杉本渉ほか4名　Fペンセット（埼玉県）柿沼輝人ほか4名　9 Aラプラスの魔（X1）（神奈川県）中村圭介　Bラプラスの魔（X68000）（愛知県）成松鋭一　Cディスクホルダー＆シール（東京都）秋山英充ほか4名　Dテレカ（愛知県）出口賢次ほか2名　10 銀河英雄伝説II（鹿児島県）林貴裕ほか1名　11 マーブルマッドネス（広島県）黒川哲也ほか2名　12 A GUNSHIP（茨城県）飯泉成弘ほか4名　B GUNSHIPポスター（長野県）丸山智ほか9名　CF-15ポスター（茨城県）三好正義ほか9名　13 Aナイフ＆ハサミセット（青森県）棟方正治ほか2名　Bフラッグディスペンサー（埼玉県）加藤健二ほか2名　14 All in Noteの世界（東京都）工藤博行ほか4名　15 Aテレカケース（兵庫県）川崎修治ほか9名　B Tシャツ（兵庫県）宮崎直也ほか4名　Cボールペン＆シャープペン（富山県）指中芳夫ほか9名　16 おみやげせっと（東京都）井上綾子　M モニタプレゼントHDX-140（新潟県）東条力

以上の方々が当選されました。おめでとうございます。商品は順次発送いたしますが，入荷状況などにより遅れる場合もあります。また，公正取引委員会の告示により，このプレゼントに当選された方は，この号の他の懸賞には当選できない場合がありますのでご了承ください。

「ひらけ！　ポンキッキ」という番組が朝の8時からフジテレビで放送されている。もちろん幼児番組であり，僕はほとんど見たことはなかった。

さて，この番組の最後の5分に，外国製の「機関車トーマス」という短いフィルムが放映されているのだが，なんでも子供たちの間でえらいブームになっているのだそうだ。テレビだけでなく，ビデオが発売されたところ，こちらもえらい売れ行きで，なんでもフジテレビの番組ビデオの中ではあの「東京ラブストーリー」に次ぐ人気だという。

試しに見てみると，顔のついた機関車が活躍するというお話だ。外国番組らしく，ユーモラスな顔なのがチャームポイントなのだろう（注，関係ない話だが，この線でいけば，新幹線はウルトラ兄弟に似ていたりする）。主役のトーマス君をめぐるいろいろなエピソードが展開され，いかにも子供向けらしい教育的内容であるので，親御さんも安心できそうな番組である。

ちょっとトレースをしてみると，人気が目に見えて高まりだしたのは今年に入ってからだが，特にここ数カ月がすごいという。別に「ひらけ！　ポンキッキ」自体の視聴率が跳ね上がり，この機関車トーマスが引きずられるかたちで人気を集めたわけではない。まったく逆で，機関車トーマスに人気があるのである。これはビデオ版が売れていることからもはっきりしている。

まあ，子供向け番組というのは，時として異様なヒットをすることがあるので，そのひとつだろうと考えれば，この話はそれでオシマイなのである。

だが，ちょっと突っ込んで考えてみると，面白いことに気づく。機関車トーマスの放映時間は午前8時20分から25分までの5分間。ここ数カ月で突然人気が跳ね上がったということは，その分，視聴率が激減している番組がなくてはならないのだが……。こう眺めると，探す必要すらなかった。つまり，NHKだ。

3月までやっていた連続ドラマ，「京，ふたり」は視聴率45％という怪物番組だった。「東京ラブストーリー」なんぞ，足元にも及ばない。僕はずっと見ていたが，連続ドラマの王道を行くような，「明日に続く」というつくりになっていたこともあり，抜群の面白さであった。

で，4月からは「君の名は」。ご存じのとおり，平均視聴率は30％以上がやっと。少なくとも10％もの視聴率が浮動票となってしまったのである。これが即，機関車トーマスに流れ出したとはもちろんいえない。だが，原因のひとつになっていることだけは間違いないだろう。

◇

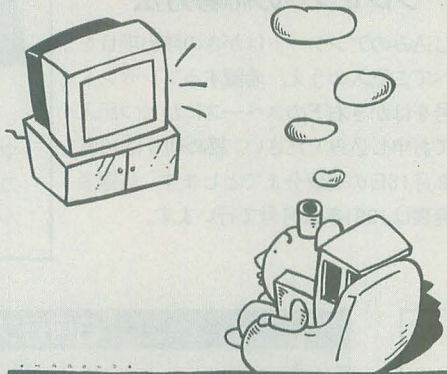
ところで，ビデオソフト需要が停滞しているそうである。ビデオソフトの関係者に話を聞くと，2人にひとりは「不景気だ」だの，「ビデオ人気もこれまでか」だのとの大合唱。

理由はいろいろあるが，まずはビデオソフトの仕入れ先であるレンタルビデオ店の

X－OVER・NIGHT

（クロスオーバーナイト）

［第14話］

ビデオ時代の転換期

TAKAHARA HIDEKI 高原 秀己

転廃業がこのところ一気に進んだこと。なんでも3年前は15,000店以上あったのが，今年に入って急ピッチで減り，10,000店ちょっとにまで落ち込んでしまったようだ。

もともと，軒先を並べるように同じ場所に2店も3店もあったところが“適正規模”に減っているので，不思議でもなんでもない。しかし，なかにはそういう競争には勝っていたほうの店が，経営に疲れてやめてしまうケースもあるようで，一般的にいう過当競争の結果というのとは，必ずしも同じではないようだ。さらに，不法レンタルをアングラでやっていたところが店じまいするケースも多い。

ソフトの動向の変化もある。新作ビデオソフトはあいかわらず多い。過去の映画はどんどんビデオソフト化が進んでいる。新作は最近のミニシアターブームもあって，洋画の小品が実に多い。さらに，テレビ番組も次々とビデオ化されているし，オリジナルビデオ（アニメだけでなくドラマでも）も続々と登場している。合計すると膨大な数だ。

じゃあ，目玉商品が多いのかというと，実は逆に減っている。劇場用映画のヒット作は年10本に満たないし，過去の大作，話題作映画はすでにビデオ化が終わっている。品数が多いわりに，人気商品は減ってきているわけだ。これはアダルトビデオでもいえる。K.K.さんに始まって，K.H.さん，S.A.さん，Tさん，M.K.さんだのといった，超人気出演者はいなくなってしまった。ビデオショップ全体として，アダルトビデオの扱いが落ちてきたこともある。飽きられてきたのだろうか。

作るほうは粗製乱造，売る側は適正水準に減って，となると全体でぱっとしないのはしごく当然だ。ここに追い打ちをかけるように，エンドユーザー向けの売り切り商品である「セルスルー」への転換というのも徐々に出てきている。15,000円ソフトのレンタルが大部分だったのだが，セルスルー用ソフトは5,000円以下。ビデオソフト流通の構造自体も変わってくる。実際に書店やコンビニでビデオソフトが売られるケースも増えてきた。

あきらかにビデオ市場自体が大きく様変わりする時期に来たのだろう。

だが，レンタルビデオ店が減ったとはいっても，密集地への過当出店が減っただけのこと。書籍，文具80,000店，自動車50,000店，コンビニ40,000店と比べると，レンタルビデオ店10,000店は決して多すぎる数字ではない。

都市部商業地に偏っていたレンタル店舗も，今後は地方や郊外地（道路沿いや住宅地など）に分散して拡大していくことは確実だ。自動車販売店が50,000店以上もあるのだから，まず間違いない。

とはいえ，衛星放送やケーブルテレビもようやく人気を集め出しており，こちらの動向の影響も気になるところ。また，パソコンやファミコンとの利用者の時間の取り合いも，今後の動向を占ううえでは無視できないだろう。

ポンコツ計算機を売る法,あるいは今世紀最後の教科書

計算機は速い，ゆえに計算機在り

もちろん，もっとすばらしい計算機，あるいはマイクロプロセッサを開発することができたのならば，何もいうことはありません。がんがん売って儲けてください。でも，もしあなたが大枚をはたいて作った計算機が思ったより遅く，すでに売られている計算機たちよりも魅力がなく見えるならば，つまりはっきりいえばそれがポンコツであったのならば，どうしたらいいのでしょう。

事実，遅い計算機など無用の長物です。現在この世に存在する計算機など人間の知的情報処理（といっても実はとるにたらないものなのだが）に比べたら，「質的」にはろくなことはできません。ただ人間にくらべて優れているのは，足し算や引き算あるいはそれに毛が生えた程度のことだけに関しては，人間など到底及ばないような速度でできるということだけなのです。

もう少し時がたてば計算機も，どういう処理ができるかという「質的」な処理で競うときがたぶんやってくるでしょう（この連載もそのような「お茶目」な計算機が前提です）。しかし，今は速いということが，特に計算機を売る局面において問われる必要十分条件なのです。したがって，あなたが作った計算機が遅かったということは，このエコロジーに対して責任を持つべきこのご時勢において，大きな罪を犯したことを意味するのです。売れないのならば。

本当のものさし

複数の計算機の速さを比較する方法はきわめて簡単です。走らせたいプログラムを持ってきて，両方のマシンで走らせてみて，実行開始から終了までの時間を測ればいいのです。もし，それがUNIXマシンなのならば，timeコマンドで引数にそのプログラム名を指定すれば，計算機が自分でそのプログラムの実行時間を報告してくれます。正確にいうと表示される最初の数字と2番めの数字を足した秒数がプロセッサを使用した時間ということになります。

走らせるプログラムがこれとこれというように固定されていて，しかも数が多くないのならば問題は何もありません（実はこれがあるのです。詳しくは述べませんが，相加平均か相乗平均かなどという問題です）。しかし，計算機を売りつけるには，どのような種類のプログラムを走らせても速いのだといわないと，あまり買ってくれません。

そこで，実行時間を計算機に固有なパラメータやプログラムに関するパラメータなどに分解する必要が出てきます。それが次の式です。

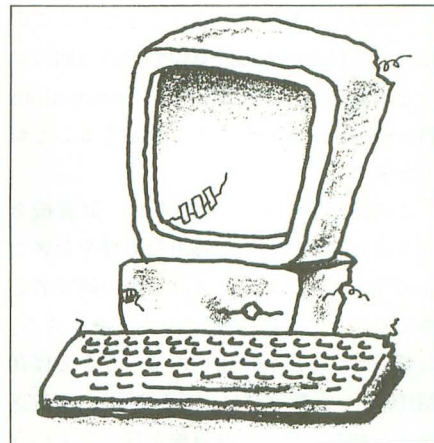
CPU時間＝命令数×1命令当たり平均使用サイクル数×クロックサイクル時間

たとえば，ある計算機のクロックが10MHzとしましょう。クロックサイクル時間はこれの逆数ですから，10ナノ秒となります。あるプログラムを実行したときに，10万個命令を実行したとします。そして1命令実行するのに平均10サイクルかかったとします。とすると実行時間は上の式に当てはめれば，

$$\text{CPU時間} = 100,000 \times 10 \times 10 \times 10^{-9}$$
$$= 10^{-2}\ (0.01\text{秒})$$

というわけです。

この式は理解してみるときわめて自然で当たり前のことを表しているということがわかるでしょう。評価するためだけでなく，新しい計算機を設計するときにはいつも念頭に置いておかねばならない重要な式であるともいえます。ただ，残念ながら，3つのパラメータは互いに完全に独立というわけではないということを忘れてはいけません。

ところで，最近流行りになってきたスーパースカラ型プロセッサというのは，2番目のパラメータを小さくするために生まれてきたものであるということがいえます。つまりRISCタイプのプロセッサは1命令当たりのサイクル数を極限まで抑え込むというアプローチに基づいて作られています。しかし，この先さらにこの数字を小さくす

るには，複数の命令を同時に実行する以外にはないことは明らかです。このような発想から生まれたスーパースカラ型プロセッサでは，複数命令の同時実行により，1命令当たり平均使用サイクル数を1より小さくすることが可能となったのです（たとえば，すべての命令が1サイクルで実行できて，しかも2命令並列実行可能ならば，この値は0.5となります）。

MIPS値でたぶらかす

計算機の速さのものさしについて説明をしましたが，たいしてむずかしいことではありません。式自体が簡単ですし，結局は速い計算機はどうしようとも速いし，ポンコツ計算機はいつまでたってもポンコツであるということです。

しかし，……重要なことは，今この世の中，ブッシュとサッチャーというご時勢です(後者はやめたようですが)。実は，このようなごく当たり前で正確な計算機の比較などまるでなされていないのです，特に，計算機の売り出し，宣伝，あるいは，雑誌の評価記事などにおいては。ということは，あなたは悲観的になることはまるでないということなのです。このような状況ではポンコツ計算機であろうが速い計算機であろうが，たいして問題はないのです。

ポンコツ計算機を魅力のあるすばらしい計算機として売り出す魔法のアイテム，それがMIPS (Meaningless Indication of Processor Speed，プロセッサの速度を示

す意味のない数値）ということです。この値は，1秒間にその計算機で何百万回の命令を実行できるか（Million Instructions Per Second）ということで測定することができます。

このMIPSというものさしは，計算機を売り込む際にきわめて強力な影響をお客さんに与えます。たとえば計算機の研究者でさえも，たとえば『日経エレクトロニクス』に載っている広告の中にある，たとえば70MIPSという数字だけに惑わされて，（貧乏かもしれないのに）計算機を買ったということも案外少なくないようです。

MIPSというパラメータの最大の長所は，先に示した計算機のCPU時間の式の中の3つのパラメータのうちの最初のパラメータである命令数にまったく影響されない点です。たとえば，同じプログラムを実行するのに，MIPS値は計算機Aが計算機Bの2倍の値だとしても，実行された命令数が3倍になっているかもしれないのです。とすると，実行時間は計算機Aが計算機Bより50％も大きくなっているのです。

このような現象を利用しない手はありません。ポンコツ計算機のクロックはとにかく速いほうが望ましく，命令セットもRISCタイプ，要するにごく基本的な命令ばかり用意されていることが望ましいのです。実行される命令数などにかまう必要はありません。

ところで，あなたのポンコツ計算機には浮動小数点コプロセッサなどはついていないでしょうね？　それはよかった。浮動小数点コプロセッサはMIPS値を悪くします。なぜならば，そのような処理は基本的で時間のかからない命令群でエミュレートしたほうがMIPS値が稼げるからなのです。

また，大事なことのひとつに，コンパイラに賢いオプティマイズ（最適化）をさせないようにするということがあります。賢いコンパイラは加算減算やシフトなどの基本的な演算命令の4割を除去してしまうことができるかもしれません。しかし，分岐命令やメモリとの転送命令のように比較的時間のかかる命令の数はあまり減らせなかったとします。これはMIPS値を悪くしてしまうのです。なぜならば，このような最適化は1命令当たりの平均クロックサイクル数を相対的に大きくしてしまうからです。

いろいろやっても，あなたのポンコツ計算機ではよいMIPS値が得られないかもしれません。それでも気を落とすことはまったくありません。ここで登場するのが「ピークMIPS値」という，これまた（買う人にとっても売る人にとっても）魅力的な（うさんくさいという言葉に置き換えてもよいが）ものです。

このピークMIPS値というのは最大瞬間風速というイメージを持たれているようですけれど，実際は現実的な使用においてそのようなMIPS値が出る必要はありません。この数字は要するに，「これ以上の性能はたとえ何が起きても出ませんよ」ということだけを表しているのです。要するに，

ピークMIPS＝青天井

なのですが，これを知っている人がまだ少ないことは幸いなことといえましょう。

実例をあげましょうか。1989年に発表されたあるマイクロプロセッサは150MIPSだとアナウンスされました。しかし，実はそれは特定の命令の並びがあったときだけのことでした。そして実際は30MIPS程度の性能しか出なかったそうです。

とにかく，速く実行できる命令パターンがあればその瞬間のMIPS値（以上）を大々的に宣伝すればよいのです，場合によっては良心が多少痛むのを少しがまんする必要があるでしょうが。

ベンチマークでペテンにかける

MIPS値でいい値を出せたのならばもうひと安心です。でも，もしかしたら，あなたのポンコツ計算機は，ほかの計算機の出しているような華麗なMIPS値が出せないかもしれません。実はそれでも，あなたは心配することはないのです。ベンチマークテストという便利なものがあるからです。でも，ベンチマークテストでいい値を出すには，優秀なコンパイラ屋をあまり安くないお金で雇わなくてはなりません。なぜならば，一種超絶的な技巧を必要とするからです。

ベンチマークテストはMIPSよりももっと信頼性が高そうに世間では見られています。コンパイルにかかる時間や出来上がったオブジェクトコードのサイズなどを気にする人はまだごく少数ですから，はっきりいうと「何でもありあり」の世界です。

コンパイル時にやれる計算，つまり入力データに依存しない計算は全部やってしまいましょう。関数を本当に呼び出さずにそのコードを呼び出す部分にそのまま埋め込んでしまいましょう。また，ループだって10回や20回の繰り返しならば，本体を10回も20回も続けて展開してしまいましょう。

もちろん，ベンチマークテストだって，実際の大きなプログラムでやられてしまうとお手上げなのですが，まだ有名なベンチマークプログラム（たとえばWetstoneやDhrystoneなど）に対する神話が存在しますので好都合なのです。

そのようなベンチマークプログラムに対するオプティマイズは，コンパイラの作り方次第では，予想以上に効果を上げることができます。たとえば1回しか通らないようなループはループの条件判断などの命令を取ったりしてしまうなどして，なんとDhrystoneのコードの25％を取り除いてしまうことができるというのです。このことは，計算機自体はポンコツでもコンパイラ次第で何とかなるというすばらしいことを示しているよい例であるといえましょう。

バイト単位の可変長の文字列のコピーは，勝手にワード単位の固定長の文字列のコピーに置き換えるような不法な手段も使うとよいでしょう。あるいはWetstoneのコードに対しては，

SQRT（EXP（X））＝EXP（X/2）

という事実を使って置き換えるとよいでしょう。ルートの計算をせずに，2で割るだけでよくなるのですから。

さらに，優れたテクニックがあります。

Arita Takaya 有田 隆也

ポンコツ計算機を売る法、あるいは今世紀最後の教科書

しかし，もうこれは実際にやっていることが公になってしまったそうですから，巧妙にやらないといけません。それは，コンパイルする前にプリプロセッサでプログラムのコメント行に書かれた著者の名前や関数の名前などをスキャンして，もしすでにわかっているベンチマークなのならば，それに対する特別の最適化（というよりはインチキ）を行えばよいのです。

ただし，あるベンチマークを実行するときだけ成績がよすぎると疑われますし，結果があまりにもよすぎるというのも考えものです。万一，知っているベンチマークと早トチリして，間違った結果を出してしまったのならば，誰も買わなくなってしまいますから注意しましょう（たまに答えを間違える計算機を買ってみたいと僕などは思いますが）。

そもそも，WetstoneやDhrystoneなどのようなベンチマークというのは，現実には決して実行されることのないプログラムです。もちろん作った人たちは種々のデータに基づいて，もっとも典型的なコードのパターンを作り出したと主張していますが。しかし，現実にはこのような威厳がありそうなプログラムこそ，こちらのいいなりになってくれるのです。せいぜいその後光を活用させてもらうことにしましょう。

いずれにせよ，いいコンパイラを作るというのはどうしても必要になります。そうすれば，あなたのポンコツ計算機は，ふだんはおんぼろ車のようでも，ベンチマークプログラムを実行するときだけは，まるでF1レースマシンのように豪快に疾走するでしょう。そのときだけ，お客さんに見せればよいのです。

「きょうの料理」風に

ポンコツ計算機を売りつける方法をあなたにそっと教えました。よいMIPS値を無理やり作りだすこと，出ないのならばよいピークMIPS値を出すことが第一です。そしてそれもだめなのなら，しかたないですから，有名でしかも（なぜか）権威のあるベンチマークプログラムにターゲットをしぼったコンパイラを作らなくてはなりません。

では，最後にこの文章がどうやって出来上がったかということをお話ししましょう。それは簡単です。次のようにすると出来上がります。

材料

1) J.L.Hennessy and D.A. Patterson, “Computer Architecture-A Quantitative Approach”, Morgan Kaufmann Publishers, 1990.

2) マルカム・ブラドベリ（柴田元幸訳）：超哲学者マンソンジュ氏，平凡社，1991.

作り方

(1)まず，材料1の第2章「Performance and Cost」を3時間ほどさらっと目に入れます。入れ方がわからない場合には英和辞典を参照してください。

(2)目の中から頭の中に材料を移し，そこで3時間ほど煮詰めます。

(3)ここが肝心ですが，材料を正確にひっくりかえします。さらに説明を加えますと，「should not」とあれば「should」に，「should」は「should not」に，あるいは「good」は「bad」に，「bad」は「good」にするなど，特に価値観に関する材料を180度丹念にひっくりかえします。

(4)ここまで調理が進むと近所迷惑になるほど匂ってきますが，ここで，文献2を隠し味として，ほんのちょっと加えます。ただし，これはお好みに応じてで結構です。

メモ

材料1は計算機アーキテクチャに関して徹底して定量的に解析した教科書として大絶賛を浴びており，出版されてからあまり時間がたっていないのにバカ売れしています。計算機アーキテクチャ関係の本としてはもう今世紀にはこれを超える本は出ないだろうとまでいわれています。それが，この本が20世紀最後の計算機アーキテクチャの教科書と呼ばれるゆえんです。

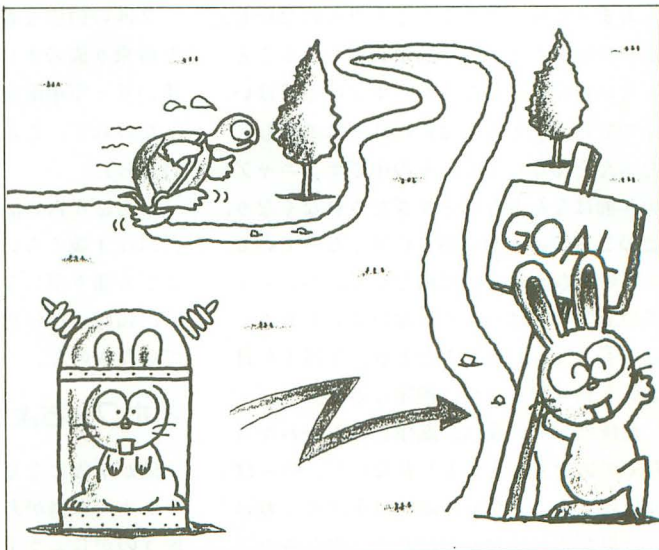
そして，新しい計算機の設計現場で，学会発表の質疑応答のなかで，あるいは，沖縄に学会の研究会でリゾートホテルに行ったときの風呂の中でまでも，この本についての話題が交換されています。

著者はRISCを提唱した人としてきわめて有名です。この本がオタッキーであるといわれる小さなエピソードをひとつ紹介することにしましょう。

この本は2人の共著ということになっていますが，書いた分量は1ページの差もなくまったく同じであると本の冒頭部で宣言しています。

さらに，あきれたことに，平等のために本書中や広告などで2人の名前が出るときも著者の並び方は，半分が片方が先で半分が片方が先になるようにしているということです。実際，表紙や本文中の記述などでかわるがわる名前の順番を入れ替えています。

そういうわけで，僕がここに引用したときも，どちらの著者を先にするか3秒ほど迷ってしまいました。

第61回

猫とコンピュータ

FAX見つけた！

Takazawa Kyoko
高沢 恭子

キョウコさんちにきたときにはまだ小さくてやんちゃだったホンニャアも，いまではりっぱなオトナのネコ。行動も落ちついて風格すら感じさせます。とはいえ，やっぱり新しいものには興味があるらしく……。

S市時代のホンニャアの親友で，はじめ女の子とまちがえてアタックしたこともあった，あの愛らしいミミが，車にひかれて死んでしまったそうだ。

S市での隣家，皮膚科のハセガワ先生によると，かつてのボス，アライグマも姿を見せなくなったという。わが家の初代の飼い猫ホンニャアは，思いがけず長生きをしているようだ。

ほんとはニャン歳？

ホンニャアが何歳なのか，このごろでは誰もちゃんと数えていない。でもたぶん7歳だと思う。

子猫のときわが家にもらわれてきたので，それからの年数をホンニャアのおよその年齢としてきた。はじめの3年くらい，それはとてもかんたんだったが，だんだん年数がふえていくと，そのたび指を折って「何歳と何カ月」とやるようになった。ちゃんと数えてもすぐまた変わるから，「何歳だろう」と考えるたびに数えなおした。

猫の年をいっしょうけんめい数えてみてもあまり意味のあることとも思われないし，誰かにホンニャアの年齢を質問されることもないのなら，私たちが「ホンニャアはいくつだろう」と考えなければそれですむ。たぶんそんなことで，家の中でホンニャアの年齢はなんとなくとりざたされなくなり，このごろでは不明の感じに近くなっている。

年齢が数えにくい理由として，ホンニャアの誕生日がはっきりしないこともある。わが家にもらわれてきたとき，生後1か月という感じだったが，推定なのだ。

血統書のない猫は，誕生日なんかわからなくてあたりまえかもしれないが，やっぱり仮定でもよいから，誕生日を決めておけば計算が明確になっただろう。ホンニャアは誕生日を持たないために，年を数えるたびにあいまいですまされてきた。

そのあいまいが何回かくりかえされるうち，ますますボンヤリとあいまいになり，こうして堂々といいかげんになったのだ。

そういえば，新宿のおばあちゃんは大正の生まれで，誕生日もハッキリしているのに，私は彼女の年齢をじょうずに数えられない。先日も実年齢より1歳多く言ったとひどく叱られた。

私は理由として，大正から昭和へのオーバーラップの部分や，誕生日以前か以後かのややこしさをあげたけれど，ゆるしてもらえなかった。誕生日のありなしでなく，いいかげんはただのクセかもしれない。

ところでホンニャアが7歳とすると，こんなに長生きの猫と暮らしたのは，これが初めてだ。子供のころに実家にいた猫たちも，せいぜい4歳くらいだった。

美人で気位が高かったメス猫のクロは，おばあちゃんの教え子の小学生が，プレゼントだといってもってきた2匹の白猫に嫉妬して，3歳ころ家出してしまった。

2匹の白猫も若く病死し，そのあとにきた肩乗り猫のチロは，飛びつこうとした相手の父と空中衝突したり，波乱の大活躍だったけれど，これも3歳くらいで行方不明になった。

いちばん古い記憶にある三毛猫のミーがたぶん4歳くらいまで生きただろうか。いまでも語り草になるほどのかしこいメス猫で，猫嫌いだった私の祖母にも気に入られていたものだ。

ボクもちょっと……

猫が7歳になると，見たところさすがにオトナの風格がある。子猫のころから人を使うのがじょうずだったけれど，ゴハンのさいそくや，外出時にドアを開ける命令など，貫録じゅうぶんだ。こうなるとホンニャアという名前に，あらためて反省もわいてくるものの，突然，小虫に飛びついたり，あいかわらずダンボール箱やタンスの引き出しに飛び込んで確認をするのを見ると，やはり本質は変わらないのだとも思う。

人間の場合，歴史上の人たちは成長に従って呼び名が変えられていったけれど，あれはなかなか楽しみもあったと思う。いまは戸籍の整理上めんどうなことはやめられて，はじめから完成された人間像を目ざして命名される。だから，おとなになってもふさわしい立派な名前だ。

猫の改名は人間にくらべたらずっとたやすいのだから，成長のようすを見ながら，マリオとかアルゴとか，つぎつぎ変えてみるのも可能だと思う。彼らは想像以上に反応が早いから，すぐに自分の名前だと納得するだろう。

1歳半くらいのころ，ホンニャアはプリンタから送り出されてくるプリンタ用紙にじゃれて，その中にいっしょに入り込んであばれて困ったものだ。

いま7歳になったホンニャアに，FAXの受信ですべり出してくる用紙の動きを見せている。彼は用紙がロールアップされてくる低いウナリ音に少し警戒して，耳を微動させながらもきちんと正座し，じっと見守っている。

新顔の機械が何カ月か前にきたのは知っていたホンニャアだったが，電話の一種だと思ってあまり気にとめないでいた。それがたまたま，FAXのとなりで午後のひる寝をしていたために，電話のベルで眠りをさまされたあげく，中から出てくる長ぁい紙を見つけたのだ。

長い長い送信だった。S市にある夫の会

社の研究所から，化学製品に関するこまかな分析データが延々と送られてきていた。

金沢に出張している夫が，本社では報告を受けられないために，帰宅したらすぐ目を通せるようにと送信されてきたものだった。

送信は1度ではすまないほど大量だったので，何通話にもおよび，とちゅうでベルが何度も鳴ることになったが，ホンニャアはさすがにおとなの貫録で，その音におびえたりすることもなく，たぶんまたズルズルと音をたてながら出てくる紙のうねりを待っていた。

たいせつな内容だから確実に受けるようにと夫に念をおされていたので，万一ホンニャアが手を出すようなことがあっては困る。私もホンニャアといっしょに並んで見守ることになったけれど，落ちついたホンニャアのけっこうりりしい横顔は，もうそんな幼稚な心配は無用だと言っているように見えた。プリンタ用紙にくるまって格闘し，紙をバリバリと引きちぎっていたホンニャアはもういない。

そう，ホンニャアはとても落ちついていた。落ちつきながら身を乗りだして，ボタンのひとつに手をのせた。送信中の用紙は半分白紙のところで停止した。

ホンニャアはスッと手をもどし，くるりと向きを変えて部屋のすみにいき，知らん顔で寝そべった。

私はもちろん大あわて。研究所のほうに電話をしてみようかと思うけれど，悲しいかな電話とFAXが共通回線。こちらが電話をするとFAXの回線は話中になって，相手が送信できなくなる。

ややあって，ふたたびベルが鳴り，つづきの送信が行われ始めた。と，数十秒もしないうちピーッとアラームの音と赤いランプ。表示窓に用紙がなくなった旨のメッセージが出た。

マズイ！　用紙の買い置きなんてないのだ。

みごとなシンプル

「FAXはありますか？」と聞かれることがたまにあっても，「あ，入れてないんです」と答えていた。通信技術の中では，電話のつぎに普及率の高いのがFAXだそうだが，いままではどうしてもFAXでなくてはならないと思うことが，あまりなかったのだ。

とくにパソコン通信の恩恵にあずかるようになってからは，FAXがとても気のきかない，一方的な電送写真のように思えたりした。それでも，折にふれて，FAXはあったほうがいいかもしれないとは，夫と話しあっていた。

FAXを入れるきっかけになったことはいくつかあった。

原稿を送るとき，いつも編集部とパソコン同士を臨時につないで，パソコン通信でデータを送っていた。あるとき，編集部の回線の調子がどうも悪くて，原稿を送ることができなくなった。

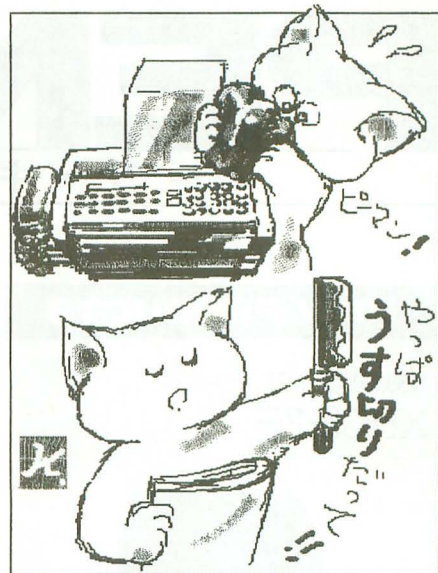
原稿は翌朝，ディスクを郵送してすませたが，少しでも早く届けたいというこんなとき，とりあえずプリントしたものだけでも送るには，FAXがいちばん確実だと急に思い始めたのだ。

そうなると，不特定の相手にアテもなくメッセージを送っていたパソコン通信が，なんだかとても無益なワザに思えてくるからおかしい。FAXなら，マシンが故障したときだって，ハードの得意なお友だちに，図入りで質問状も出せるじゃないか。しかも，アクセスしなければ自分へのメールのありなしがわからないという，間のぬけたこともない。

夫もしごとなどで，FAXの必要を感じることが多くなっていたので，話はすぐに同意を得られた。

パーソナルファクシミリだというこの機種は，電話と共用できるタイプで，番号登録や短縮ダイヤルをはじめ，パスワードによって相手を指定できるセレクト受信，一部，外出先からも遠隔操作のできる機能など，パソコン感覚のマシンだ。もちろん留守番電話も接続できるし，ほかにも親切すぎる機能がたくさんある。

FAXがこんなに便利だということは，使ってみるまでわからなかった。その便利さを実感したのは，目的がシンプルで確実だということだった。書いたものが，そのままの姿で，瞬時に遠方に送られる。活字でも，手書きでも，絵でも，いまここにあるものが離れたところに復元される。

通称FAXはファクシミリ，もとは書籍や絵画などを本物そっくりに複写，複製する意味だそうだ。決められたところに，文章

や図表，絵などを電送するだけの目的とはいえ，そのシンプルさに拍手を送りたくなる。

これはもともと，パソコン通信とは別の分野での通信手段なのだ。

ヒミツはきらい

ところで，そんな便利さに感動していたFAXだが，受信するときには当然のことながら，記録用紙を消耗する。用紙は機械の内部にあるし，そんなにひんぱんに受信するわけでもないので，残りの量についてあまり注意しないでいた。でも，きょうの研究所からの送信はあまりに大量すぎた。

用紙がなくなってしまっては，こちらから連絡しなくてはならない。

「もしもし，あの，じつは……」

と言いかけたら，秘書課のモリヤさんは，

「すみません，こちらのFAXが先週から故障がちでして，何回も中断してしまうんです。あの，どのあたりまで届いておりますでしょうか……」

そこで，あらためて書類に目を通してみると，同じ内容のものがエンドレスで循環して送られている。

ホンニャアが手を出して中断したと思ったのも，送信側のトラブルだったようだ。

FAXの欠点もある。秘密の伝達がとてもむずかしいということだ。ズズズーと音をたてながら，さあごらんくださいとばかりに，おおらかにせり上がってくる。

いま，いちばんうれしい受信は，倉敷のナカタさんのお嬢ちゃん，2年生のユカちゃんから，絵入りで届くお手紙だ。

PENGUIN INFORMATION CORNER

ペ・ン・ギ・ン・情・報・コ・ー・ナ・ー

NEW PRODUCTS

X68000用3.5FDドライブ

X6835-2F

ファーベル

X6835-2F

ファーベルでは，X68000用の3.5インチフロッピーディスクドライブユニット「X6835-2F」を発売した。

「X6835-2F」は2ドライブを搭載していて，扱えるメディアは3.5インチの2HDのみとなっている。

Human68k上ではそのまま動作可能で，OS-9/X68000ではデバイスディスクプリタ(標準添付，5インチ2HD)で対応する。

情報転送速度は500KBits/sec。外形寸法は幅104mm×高さ110mm×奥行き190mm，重量は約1.5kg。

価格は80,000円（税別）。

〈問い合わせ先〉

㈱ファーベル　☎092(512)3661

書院新4機種

WD-A520/540/550/560

シャープ

シャープは，書院スーパーアウトラインフォントの搭載に加え，光通信コードレス10キー「10キーステーション」を装備したラップトップワープロ「WD-A550/560」，個性で選べるパールカラー&ラウンドフォルムのラップトップワープロ「WD-A520/540」を発売した。

「WD-A550/560」が搭載している10キーステーションは数値の入力だけではなく，カーソル移動，書式設定，移動，複写，アンダーライン，倍角などといった基本的な編集も行える。この10キーの採用により，「ひらがな入力」を使用している場合でも，

WD-A560

英数モードに切り替える必要がなく，手間を省くことができる。使用中以外は本体内に収納できるので，持ち運びするときにも便利である。

また，今回発売したなかでの最上位機種「WD-A560」では，毎秒93文字の高速印刷が可能。さらに，「感熱紙専用ドラフト印刷」では印刷ヘッド進行方向の解像度を1/2にすることで毎秒140文字の高速印刷が可能になった。表形式の文書で威力を発揮する。

「WD-A520/540」は本体に滑らかなデザインで形成したラウンドフォルムデザインと，個性的なパールカラー「エレガンスパールホワイト」，「スポーティカジュアルブルー」を採用している。

そのほか，すべての機種に新たな機能，「らくらく自動表計算」が搭載されており，「合計」，「平均」などの項目名と数値を入力するだけで計算と罫線引きを自動的に行ってくれる。

表示画面は「WD-A520」のみがブルーモード液晶で，そのほかの機種はハイコントラスト白黒液晶。価格は「WD-A520」が178,000円，「WD-A540」が198,000円，「WD-A550」が208,000円，「WD-A560」が248,000円となっている（すべて税別）。

〈問い合わせ先〉

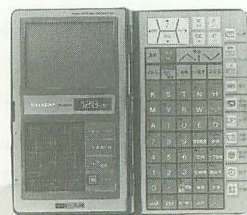
シャープ㈱　☎03(3260)1161，06(621)1221

大容量ハイパー電子システム手帳

PA-9550

シャープ

シャープは，現在発売中のハイパー電子システム手帳「PA-9500」の上位機種として，電子手帳では最大の本体RAM容量128Kバイト，そして，データの整理，編集に便利なカット/コピー/ペースト機能などにより，より使いやすさを追求した，「PA-

PA-9550

9550」を発売した。

本体に128KバイトのRAMを搭載することで，電話帳なら約2260人分（名前全角4文字，読み4文字，電話番号半角数字12桁の場合），スケジュールなら約1650件(年月日，開始時刻，終了時刻）登録することができるようになった。

また，カット/コピー/ペースト機能は電話帳から名刺管理，電話帳1から電話帳2などで使うことができ，データを整理するときに便利な機能である。

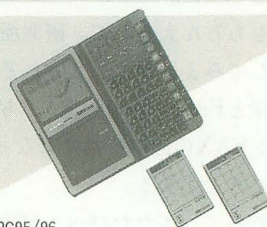
価格は59,000円（税別）。

〈問い合わせ先〉

シャープ㈱　☎03(3260)1161，06(621)1221

電子手帳用プログラムBASICカード

PA-9C95/9C96/9C97

シャープ

PA-9C95/96

シャープでは，ハイパー電子システム手帳用ICカードとして，高速演算を実現し，ハイパー電子システム手帳の大画面，ディスプレイタッチパネル対応のBASIC命令を搭載したプログラムBASICカードを新開発，8月から発売する。

従来より業種，業態に合わせてBASICで専用のICカードが作成できる「PA-7C18/19」が商品化されていたが，今回発売のプログラムBASICカードは従来モデルに比べ約18倍の高速演算を実現。また，漢字12桁8行のフルグラフィック画面（192×145ドット）を使ったプログラム作成ができ，ディスプレイタッチパネルのメニューやア

イコンなどの定義などを行うBASIC命令も搭載している。

使い勝手の面も，電子手帳上でプログラム入力，修正が可能なエディタ機能や，プログラム実行中に自由に電子手帳機能が利用できるプログラムリジューム機能などの採用により，充実している。

現在のところは容量64Kバイトの「PA-9C95」，128Kバイトの「PA-9C96」のみのラインアップとなっているが，12月からは受注生産により640Kバイトの「PA-9C97」も販売する。

プログラムBASICカードの価格はオープンとなっている。また，同時に開発用ツールとして，パソコン上でプログラム作成を可能にするプログラムBASICカード開発マニュアル（ソフト同梱）「CE-150D」，および，通信ケーブル「CE-150T」も発売される。こちらの価格はともに10,000円(税別)。

〈問い合わせ先〉

シャープ㈱　☎03(3260)1161,06(621)1221

電子手帳用ゲーム

PA-3C34/3C36

ココナッツ21，ビクター音楽産業

PA-3C34/36

シャープの電子システム手帳用ICカードとして，以下の2機種のゲームが発売された。

○ザ・ベースボールカード

テキストアドベンチャー形式の新しい野球シミュレーションゲーム。アクション性をできるかぎり削除して，データによる頭脳プレイに重点を置いているので，野球ファンなら誰にでも楽しめるゲームになっている。チームはセ，パ両リーグ12球団を用意。投手は防御率，球速，球種，打者は打率，長打率など，詳細な216人分のデータが入力されている。また，打率，防御率の変動，グラフィックの挿入などで，臨場感にあふれている。パスワード機能により，ゲーム途中からの再開も可能。

価格は9,800円（税別）。

〈問い合わせ先〉

㈱ココナッツ21　☎03(3288)5563

○フォートレスカード

米国ジョージ・ワシントン大学ジェイムズ・テンプルマン博士考案の，囲碁とオセロを合わせたような新しいタイプのゲーム。白と黒の領主が領地を増やすために争う。人間対コンピュータはもちろん，人間同士，コンピュータ同士の対戦も可能。「ニューラル・ネットワーク学習システム」を搭載しているので，コンピュータは対戦するたびに学習して強くなっていく。

価格は7,800円（税別）。

〈問い合わせ先〉

ビクター音楽産業㈱　☎03(3423)7901

X68000用体験版ソフトつき

SUPER PROシリーズ

化成バーベイタム

SUPER PRO
シリーズ

化成バーベイタムでは，DataLife SUPER PROシリーズに体験ゲームソフト1枚を添付した，DataLife SUPER PROアドインゲームシリーズを発売した。

添付される体験版ソフトは，X68000用には「シューティング68K」，PC-9801用には「クリスタルチェイサー」，「AIZA」が用意されている。

〈問い合わせ先〉

化成バーベイタム㈱　☎03(3283)6423

INFORMATION

パソコン通信「EYE-NET」

「音声」，「英日機械翻訳」サービス

フジミック

フジミックでは，同社が運営しているフジサンケイパソコン通信ネットワーク「EYE-NET」において，「音声サービス」，「英日機械翻訳サービス」の2種のサービスを開始した。

「音声サービス」は電子メールとして送信された文書を，あらかじめ「EYE-NET」内に登録されている会員番号と暗証番号をプッシュホンで入力することにより，音声で聞くことのできるサービス。7月末までは実験期間でそれ以降に本サービス運用を計画している。実験期間中は利用料金は無料。利用手続きはオンラインでできる。

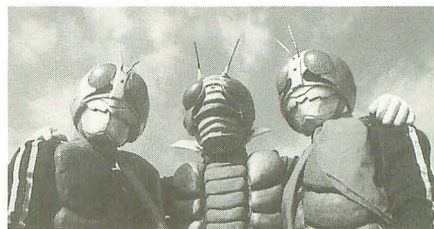
「英日機械翻訳サービス」では電子メールで送信された文書を，機械翻訳システム開発会社「ノヴァ」で英日機械翻訳システム「Transfer/EJ」を用いて翻訳しユーザーに返送する。翻訳料金は翻訳結果1文字につき3円で，A4サイズ1枚(日本語360文字に相当)の翻訳料金は1,080円となる。ただし，1回の最低料金は2,000円。

〈問い合わせ先〉

㈱フジミック　☎03(3358)0591

東映TVヒーロー

フィルム・マラソンご招待

東映ビデオ

誕生20周年を迎える仮面ライダーと不滅の人気東映TVヒーローのLDが，この秋東映ビデオから発売される。これを記念して東映ビデオでは，東映TVヒーローの作品の数々を上映するイベントを行う。

○上映作品

「仮面ライダー　世界を駆ける3D版」「仮面ライダーV3」「仮面ライダーアマゾン」「快傑ズバット」「宇宙刑事シャイダー」「マジンガーZ」「スーパージェッター」「超電磁ロボ　コンバトラーV」「仮面ライダー劇場版予告編」など（変更の場合あり）

○日時と場所

8月31日（土）　丸ノ内東映

10時開場，終了は翌日の午前5時の予定

○応募方法

官製ハガキに住所，氏名，年齢，職業，電話番号を明記のうえ，以下の住所まで。

〒104　中央区銀座3-15-10　東映ビデオ宣伝部　「東映TVヒーローフィルム・マラソン　Oh!X」係

また，当日のコスチュームプレイの参加者も募集している（東映のキャラクターに限る）。

希望者は自作のコスチューム着用の写真を同封のうえ，試写会応募と同じ要領で，“東映ビデオ宣伝部「東映TVヒーローフィルム・マラソン コスプレ参加係」”まで。

締め切りはともに8月10日（土），当日消印有効。なお，深夜上映につき18歳未満の方は参加不可。

〈問い合わせ先〉

東映ビデオ宣伝部　☎03(3545)9302

FILES Oh!X

このインデックスは，タイトル，注記——筆者名，誌名，月号，ページで構成されています。さあ，夏休み。宿題は早めにすませて，プログラミングやゲームにいそしみましょう。でも，たまには外に出ようね。

参考文献
I/O　工学社
ASCII　アスキー
コンプティーク　角川書店
C MAGAZINE　ソフトバンク
テクノポリス　徳間書店
POPCOM　小学館
マイコン　電波新聞社
マイコンBASIC Magazine　電波新聞社
LOGIN　アスキー

一般

▶NETWORK CONNECTION
メガドライブを使った，プロ野球のリアルタイムデータサービスVAN「プロ野球VAN」を紹介。ほかにネットワーカーの男女比についての討論など。——編集部・ナカジー・ウニーニ，LOGIN，11号，298-299pp.

▶IBM-PCに注目!!
テラドライブ発売でますます注目されている世界標準パソコン「IBM PC」をソフト，ハード両面から解説。——編集部，マイコンBASIC Magazine，7月号，67-72pp.

▶NEWS! シャープパソコンフォーラム'91「見体験フェア」報告
去る5月11・12日池袋で開催されたシャープのパソコンイベントの模様を紹介。XVIの紹介はもちろん，新作ソフトやユーザーのオリジナルソフトの発表，X68000オリジナルグッズの販売なども行われた。——編集部，テクノポリス，7月号，150p.

▶'91夏　最新機種がやってくる
ASCII 7月号特集。ビジネスショウなどを契機に発売された各社の最新型パソコンを一挙に紹介する。デュアルCPUのテラドライブ，ソニーの新型パームトップなども登場。——編集部，ASCII，7月号，229-252pp.

▶パソコンで体験する天文学
宇宙にまつわるいろいろなことがらをコンピュータでシミュレートしようという新連載。第1回は太陽ヨットレースとスペースコロニーでのボール投げをプログラムする。——福江純，ASCII，7月号，302-307pp.

▶最新主要機種スペック一覧
国内で市販されている主要なパーソナルコンピュータの性能の概要と価格を一覧表にまとめている。——編集部，ASCII，7月号，426-432pp.

▶第72回ビジネスショウ
5月15日から5月18日にかけて晴海で開催された第72回ビジネスショウの模様をメーカーの展示内容を中心にレポートする。MS-WINDOWS 3.0関係の出展が目立ったようだ。——高橋雄一，マイコン，7月号，107-112pp.

▶マイクロコンピュータショウ'91誌上レポート
5月8日から11日にかけて東京流通センターで開催されたマイコンショウを企業別にレポート。シャープは液晶ディスプレイのほか，X68000XVIや新作ソフトを展示。——高橋雄一，マイコン，7月号，113-117pp.

▶東南アジア電脳事情見聞録
気の向くままに東南アジア諸国を見て回る旅行記の2回目。シンガポールとインドネシアを舞台にコンピュータ街の模様，普及の度合いやコピーの問題を報告する。——上野隆平，マイコン，7月号，264-268pp.

▶たのもー電脳展覧
マイコンショウ'91のレポート異聞録。コンピュータ音痴の筆者とゲーム狂の記者が見たマイコンショウの模様とは？——編集部，マイコン，7月号，418-419pp.

▶春季コムデックス'91
アメリカの大規模なコンピュータ発表展示会コムデックス'91の模様を社別の展示内容を中心にレポート。——デイナ・ブランケンホーン，I/O，7月号，214-217pp.

MZシリーズ

MZ-1500(BASIC MZ-5Z001)

▶SMALL LAND
軽いノリのシューティングRPG。平和な王国SMALL LANDを救う騎士のお話ゲーム。——大石豊，マイコンBASIC Magazine，7月号，125-127pp.

MZ-2500(BASIC-M25)

▶Little Red Riding Hood II　～バレーボール編～
2人用バレーボールゲーム。——BMN，マイコンBASIC Magazine，7月号，128-129pp.

X1/turbo/Z

X1シリーズ

▶チェッカーフラッグ
2人同時プレー可能のレースゲーム。ポツ続きの少年が作った!? ショートプログラム。——堀田英克，マイコンBASIC Magazine，7月号，152-153pp.

▶愛してるぜ！　編集部
キャラクターをタテ，ヨコに並べて得点をとるパズルゲーム。——JIRONKA，マイコンBASIC Magazine，7月号，154-156pp.

X1＋音源ボード（要NEW FM音源ドライバ）

▶アクトレイザー　～平和な世界～
エニックスのスーパーファミコンのゲームより，ミュージックプログラム。——桐畑厚宏，マイコンBASIC Magazine，7月号，186-187pp.

X1turboシリーズ

▶影さんのふくわらい
マウスで顔のパーツを置いていく。アイデア福笑いゲーム。——北沢高志，マイコンBASIC Magazine，7月号，157-158pp.

X68000

▶NEW SOFT
市販のゲームじゃあきたらない君へ，ゲームコンストラクションソフト「シューティング68K」を紹介。——編集部，LOGIN，11号，22p.

▶最新ゲーム徹底解剖!!

新刊書案内

人工現実感。アーティフィシャル・リアリティの訳だ。人工現実感でピンとこなければ，仮想現実感（バーチャル・リアリティ）といってもいいだろう。その人工現実感の現状と未来について，アメリカや日本の最先端を取材して紹介した本である。

人工現実感について著者はこう説明している。「コンピュータが作り出す世界の“中”に人間が入り込むテクノロジーを指す」

全体的には，人工現実感というより，人間とより密着したインタフェイスを実現するコンピュータの姿を追求した部分が目立つ。ある意味でそれはサイバースペースだったりデータグローブだったりフライングマウス（3Dマウス）だったり馬鹿でかいゴーグル（内側に液晶スクリーンがあって，そこに画像が映る代物）だったりする。

人工現実感の流れは2つあるようだ。ひとつは産業に応用して役に立たせようとする方向。もうひとつは，エレクトリカル・ドラッグという，文化的な方向。どちらにしろ，先端の企業がやっている研究はユニークで面白い。もし体験ツアーがあるなら，積極的に参加したい。（K）

人工現実感の世界　服部桂著　工業調査会刊
☎03(3817)4701　A5判　278ページ　2,300円

書籍の価格は消費税込みです

ズーム入魂の新作「ファランクス」を，6ページを費やして紹介。そのほか「スコルピウス」，「パロディウスだ！」の攻略第2回，「ノスタルジア」など。──編集部，LOGIN，11号，154-207pp.

▶X68000新聞

移植中の「サイレントメビウス」，ますます速くなった「C-TRACE68＋TP版」と，CGコンテスト入賞作品の紹介。パズルゲーム「スターモビール」や新作情報「シューティング68K」「生中継68」「キャンペーン版大戦略II」など。PDSは，S-RAM常駐スタートアップツール「SBUILD」。──編集部，LOGIN，11号，278-281pp.

▶NEW SOFT

アクションゲーム「装甲騎兵ボトムズ」，パズル「スターモビール」，ディスクマガジン「フェアリーテール海賊版」など。──編集部，LOGIN，12号，19-25pp.

▶最新ゲーム徹底解剖!!

アクションゲーム「ファランクス」，謎の分岐点とボスの攻略法を紹介。ほかに「パロディウスだ！」「スコルピウス」「ノスタルジア」「シグナトリー」など。──編集部，LOGIN，12号，134-187pp.

▶X68000新聞

新着ゲーム「大戦略III'90」「アクアレス」「ループス」「黄金の羅針盤」「ダッシュ野郎」，期待の「Multi word」や「Teleportion PRO-68K」を紹介。「もっとX68000を活用しよう！」では，「今ネットが花盛り」と題して市販とPDSの通信ソフトや「HOST PRO-68K」を紹介。──編集部，LOGIN，12号，262-265pp.

▶NEW PRODUCTS

X68000用常駐型ユーティリティ「Teleportion PRO-68K」を紹介。その通信機能，エディタ，カレンダー，スケジュール管理ツール，住所録など。──6st.inv，マイコンBASIC Magazine，7月号，80-81pp.

▶誌上公開質問状

MUSIC PRO-68K［MIDI］とMusicstudio PRO-68Kの違い，数値演算プロセッサボードの効果，BASICでグラフィックをロード＆セーブする方法，プリンタCZ-8PC4で画面のハードコピーを取るには？ などに答える。──多田太郎，マイコンBASIC Magazine，7月号，89-90pp.

▶TACHYON

双方代表による戦争の決着。ショートプログラム2人用対戦ゲーム。──鈴木達郎，マイコンBASIC Magazine，7月号，159-160pp.

▶POWER BOMBER

アイアンボンバー（鉄球）を使った接近戦で敵をやっつける。サイドビューの1対1格闘ゲーム。──福田圭介，マイコンBASIC Magazine，7月号，161-162pp.

▶GROUND MASTER

花の咲かない大地を2人の協力で救う。2人同時しかできないアクションゲーム？──多賀英明，マイコンBASIC Magazine，7月号，163-165pp.

▶パソコン版ソーサリアン ～エンディング1～

ゲームミュージックプログラム。要NAGDRV＋MT-32系MIDI楽器。──荒木潤，マイコンBASIC Magazine，7月号，188-193pp.

▶GAMING WORLD

9月発売予定の「シムアース」や，「ザ・マジック・キャンドル」，大好評発売中の「ファランクス」「ノスタルジア」「ダッシュ野郎」など，新作情報を掲載。──編集部，テクノポリス，7月号，6-42pp.

▶SOFT EXPRESS

スターモビール，装甲騎兵ボトムズ，機動戦士ガンダム・クラシック・オペレーション，マーキュリー，ファランクス，ダッシュ野郎，シューティング68Kを紹介。──編集部，コンプティーク，7月号，76-78pp.

▶Software Hot Press

「スターモビール」「ダッシュ野郎」を紹介。──編集部，POPCOM，7月号，24-25pp.

▶ゲームの達人

「ファランクス」の徹底攻略！ 3～5面を大紹介。──編集部，POPCOM，7月号，88-89pp.

▶ミュージックパビリオン

「行くぞ大洋」横浜大洋ホエールズの応援歌より。ミュージックデータ。──編集部，POPCOM，7月号，151-153pp.

▶AVプログラミング講座

疑似3次元表示を扱う第2回。今回は立体物を表すデータの作り方と，ポリゴン表示のアルゴリズムについて。──宮本親一郎，ASCII，7月号，333-340pp.

▶AV STRASSE

シャープから発売された「Teleportion PRO-68K」を紹介する。このソフトはスケジュール管理，住所録といったデスクアクセサリ機能と，パソコン通信のターミナルソフトとしての性格を持っている。──編集部，ASCII，7月号，349-352pp.

▶FREE SOFTWARE INDEX

パソコン通信の大きな楽しみがPDSの入手。その数々のPDSのなかから，最近登場した面白そうなものを選んで紹介するページ。X68000用PDSも多数紹介。──編集部，ASCII，7月号，392-396pp.

▶HOBBY EXPRESS

キャメルトライ，ファランクス，サブナック，シグナトリーの4本を紹介。──MUNEPI♫・あゆさわかつみ・相川春利・成島月美，マイコン，7月号，347-371pp.

▶なんでもQ＆A

S-RAMの初期化のやり方，SX-WINDOWのシステムファイルにはどんなものがあるか，など。──シャープ株式会社液晶映像システム事業部第2商品企画部，マイコン，7月号，400-401pp.

▶Merry Maze

迷路の塔から水晶を取り戻すアクションゲーム。ブロックを押して道を作り，水晶を階段まで持っていけばクリアだ。──土方嘉徳，I/O，7月号，130-132pp.

▶Misnon

タイプミスの多い人に便利なユーティリティ。コマンドを入力するときに間違いがあると正しい候補を示してくれる。──（は），I/O，7月号，140-145pp.

▶SOFT BOX

CARD PRO-68K Ver.2.0を紹介する。BASICライクな機能が用意されたカード型データベース。画面センスの評価が高い。──武石秀男，I/O，7月号，182-184pp.

▶数値演算コ・プロセッサ・ボードの製作

X68000の数値演算プロセッサボードは6万円もする。これでは高い，という人のために自作の方法を紹介する。──市原昌文，I/O，7月号，206-209pp.

▶GAME BOX

イマジニアから登場するシムアースを取り上げる。移植の完成度もなかなか高いようだ。──沢崎正光，I/O，7月号，106-107pp.

▶GCCで学ぶ68ゲームプログラミング

なめらかスクロールの秘訣その2。ハードウェアに密着した，やや凝ったスクロールテクニックを紹介する。疑似多重スクロール（ラスタースクロール），割り込み処理の方法などについて解説。──吉野智興，C MAGAZINE，7月号，125-129pp.

ポケコン

PC-E500

▶ぴぴぴーぷ

ミュージックスタッフへの第一歩，音階を聞き分ける実用ゲーム。──めが，マイコンBASIC Magazine，7月号，168p.

▶LOOK AT!!

一風変わったオリジナルつぼふりゲーム。──佐藤祐紀，マイコンBASIC Magazine，7月号，169p.

PC-1262/PC-1360K

▶誌上公開質問状

PC-1262とPC-1360Kのプログラムの互換性についての質問に答えている。──ドラゴン，マイコンBASIC Magazine，7月号，90p.

PC-1600K

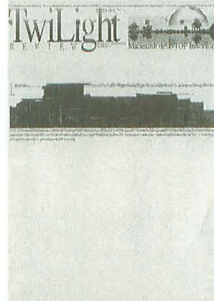
▶PC-1600K実践プログラミング

パーソナルレベルを主眼においたポケコン活用法講座。今月はマニュアル計算の活用を解説する。変数を使ったマニュアル計算，プレイバック機能など，ポケコンならではの使い方を教えてくれる。──塚田洋一，マイコン，7月号，308-312pp.

2000年のコンピュータ社会を読む

2000年のコンピュータ社会を予測するというより，2000年にどういう社会になっているといいかを希望する，というような内容だ。コンピュータだけでなく，古代ローマから教育問題まで広く扱うが，ポイントは，いままでのコンピュータに対して「ひたすら企業あるいは産業優先の効率追求と生産性向上の道具として使われてきた」と否定し，より知的能力を高めるような，人間中心のコンピュータ利用を唱えていることだ。（K）

栗田昭平著　パーソナルメディア刊

☎03(5702)0502　四六判　340ページ　1,854円

TwiLight Review 1

戸田ツトム氏がまたやった，という感じだ。すべてがMacintoshを使ったDTPで作成している。それゆえ，戸田ツトム氏のセンスや技術が詰まっており，恐ろしくサイバーでとても綺麗なムックに仕上がっている。副題に「電脳風景新世紀」と書いてあるだけのことはある。DTPと名乗るなら，ここまで行き届いたものにすべきだろう。図版の作成方法の解説や，テッド・ネルソンのインタビューがなかなか面白い。（K）

TODA office SPHINX編集制作　太田出版刊

☎03(3263)7770　96ページ　A4判　3,090円

書籍の価格は消費税込みです

QUESTION and ANSWER

5月号の黄金週間PRO-68Kに入っていた“APIC_LIB.A”ですが，C言語でプログラムを書いて“PIC_LIB.A”を使用した場合は問題なく動作しますが，ソース中の“PIC_LOAD( )”を“APIC_LOAD( )”に変更し，APIC_LIB.Aを用いてコンパイルした場合，プログラムを実行するとバスエラーやアドレスエラーが発生してハングアップしてしまいます。なぜなんでしょうか？北海道　江崎　康幸

結論からいえば，APIC.FNCおよびAPIC.LIBにバグがあったのです。あのプログラムを作成したのは実は私なんですが，自宅のPROではなんの問題もなく正常に動作していたので，安易にマスターアップしたのが間違いでした。5月号発売直後に「APIC.FNCにバグがあるのではないか？」と，読者から問い合せの電話がかなりあったようです。ご迷惑をお掛けしました。編集部でも特定のマシン(環境)でAPIC.FNCを使用すると，必ず暴走するといった症状が確認されました。

ソースリストを眺めていたところ，原因がわかったので報告します。

アセンブラを使える方は，付録ディスクから解凍して作られたディスク3に収められている3つのソースプログラムを以下のとおり変更してください。

・APIC_FNC.S
　245行に，add.l d0,d1を挿入
・APIC_LOAD.S
　124行に，add.l d0,d1を挿入
・APIC_SAVE.S
　131行に，add.l d0,d1を挿入

念のために，APIC.FNC, APIC.LIBの作成方法を説明しておきます。まずAPIC.FNCは，

```
as apic_fnc
lk apic_fnc /o apic.fnc
```

でOKです。次に，APIC_LIB.Aですが，

```
as apic_load
as apic_save
```

で.oファイルを作成しておいて，

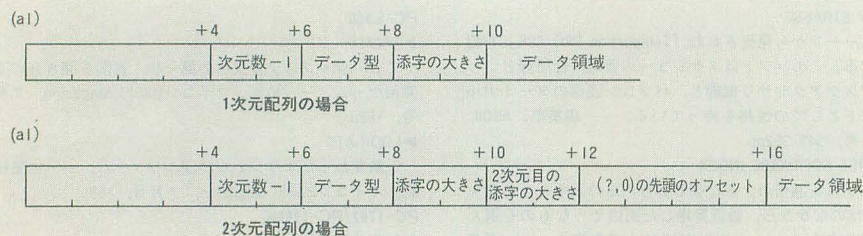
```
ar /u APIC_LIB.A apic_load.o apic_save.o
```

としてください。

X-BASICの外部関数で，引数を配列変数にしたとき，その引数はどこへ行っているのかわかりません。実例（アセンブラ）を添えて説明してください。　愛媛県　大森　亮寛

通常，X-BASICインタプリタは外部関数の引数をパラメータスタックと呼ばれる10バイトの領域に，引数の型(2バイト)，引数の値(8バイト)を格納します。しかし，引数に配列を与えた場合は事情が違います。この場合はパラメータスタックに配列へのポインタを格納するのみです。したがって引数が配列でない場合と，配列である場合では引数の受け取り方もまったく別ものになります。質問では引数がどこに行ったかわからないということですが，引数のありかはポインタの指すアドレスにあります。ユーザーはこのポインタを頼りに，配列とデータのやりとりをします。

では，ポインタはパラメータスタックのどこに格納されているのか？　これは引数の値を格納する8バイトのうち，下位4バイトに割り当てられています（上位4バイトには0が入る）。X-BASICから外部関数が呼び出されると，スタックにはリターンアドレスが4バイト，引数の総個数が2バイト積まれ，この後10バイトのパラメータ

図1

リスト1

```
 1: * infomation table
 2:
 3:         dc.l    x_init
 4:         dc.l    x_run
 5:         dc.l    x_end
 6:         dc.l    x_sys
 7:         dc.l    x_brk
 8:         dc.l    x_ctrl_d
 9:         dc.l    x_res1
10:         dc.l    x_res2
11:         dc.l    ptr_token
12:         dc.l    ptr_param
13:         dc.l    ptr_exec
14:         dc.l    0,0,0,0,0
15:
16: x_init:
17: x_run:
18: x_end:
19: x_sys:
20: x_brk:
21: x_ctrl_d:
22: x_res1:
23: x_res2:
24:         rts
25:
26: ptr_token:
27:         dc.b    'sum',0
28:         dc.b    0
29:
30:         .even
31: ptr_param:
32:         dc.l    test_par
33: test_par:
34:         dc.w    $0032   * int 型1次元配列
35:         dc.w    $8001   * int_ret
36: ptr_exec:
37:         dc.l    sum
38:
39:         .text
40:         .even
41: sum:
42: *               a1.l=配列へのポインタ
43:         movea.l 6+6(sp),a1
44:
45: *               d0.w=配列の要素数－1
46:         move.w  8(a1),d0
47:
48: *               a1.l=データ領域
49:         lea.l   10(a1),a1
50:
51:         clr.l   d1              * 総和
52: loop:
53:         add.l   (a1)+,d1
54:         dbf     d0,loop
55:
56:         lea.l   ret_data,a0
57:         move.l  d1,6(a0)
58:         clr.l   d0
59:         rts
60:
61:         .data
62: ret_data:
63:         dc.w    0
64:         dc.l    0
65:         dc.l    0
66:
67:         .end
```

スタックが引数の数だけ繰り返し積まれます。すなわち，第1引数として配列を与えた場合は，

```
movea.l 6+6(sp),a1
```

で，a1に配列の置かれているアドレスを入れることができます。同様に第2引数の場合は，

```
movea.l 6+16(sp),a1
```

となります。22(SP)としないで6+16(SP)としているのは，リターンアドレスと引数の総個数の和である6バイトをパラメータスタックと区別しているからです。こうしておくと，あとで見直すときでも一目瞭然です。

しかし，これだけでは配列の情報がなにもありません。配列の情報とは，配列の型や要素数といったものです。実はポインタの指すアドレスには，これらの配列情報も格納されています。図1にこれらの情報がどのように格納されているかまとめておきました。

では，アセンブラでの作成例を示しましょう(リスト)。リスト中のコメントを見てもらえば詳しい説明はいらないでしょう。

X68000マシン語プログラミングで発表されたラインルーチンをMAGICに組み込もうと思ったのですが，やり方がわかりません。プログラムの変更点を詳しく教えてください。

福島県　大木　明

まず，マシン語プログラミングで掲載されたプログラムのなかから必要なものを取り上げておきます。

1990年9月号
　リスト1　GRAMADR.S
　リスト2　GCONST.H
1991年5月号
　リスト7　CLIPRECT.S
　リスト8　GLCLIP.S
　リスト9　GLINE2.S

最後に挙げたGLINE2.Sについては6月号で，より高速なGLINE.S（高速版）が掲載されましたので，そちらを使用することをおすすめします。本文中の“垂直線分描画ルーチンの高速化”を参考にして変更を加えるとさらに速いルーチンになります。

ご存じのように，MAGICは機能ごとにモジュール分割して開発したので，あとから別の高速ラインルーチンを組み込んでみようというときでも，ライン描画に関係するモジュールのみの最低限の変更ですみます。そう考えると，変更するモジュールはLINE.SとDISP_FLAME.Sのように思われます。

最初に考えなければならないことは，ラインルーチンへのパラメータの受け渡し方です。IOCSコールは

```
lea.l line_data,a1
IOCS  _line
```

と，a1にパラメータ格納アドレスを指定するのに対して，マシン語プログラミング掲載版では，

```
pea.l  line_data
bsr    gline
addq.l #4,sp
```

とスタックにパラメータ格納アドレスを積んで呼び出すことになっているのです。ところで，パラメータの格納形式は両者とも同じなので呼び出し方法の変更だけですみます。

さて，DISP_FLAME.Sではライン描画のほかに描画ページの切り替えも行っています。マシン語プログラミング掲載版では独自のサブルーチンを使って描画ページを切り替えているので，これについても考慮する必要があります。

すなわち，

```
IOCS _APAGE  →  bsr apage
```

に変更します。なお，ラベルgline，apageはラインルーチンのエントリラベルです。このラベルは外部参照ですから，ソースリストの頭で，

```
.xref gline
.xref apage
```

という宣言をしておかなければなりません。

ところが，マシン語プログラミングで紹介されたラインルーチンは，ほかのグラフィック関係のIOCSコールとのつじつま合わせのために，IOCSコールWINDOWを用いてIOCSのワークにウィンドウエリアを設定してはいますが，実際のクリッピングは独自のワークに格納されたウィンドウエリアを参照して行います。ということで，ウィンドウエリアを設定するWINDOW.Sにも変更を加える必要があります。文章では説明しにくいので実際の変更点を図2にまとめておきました。

図2

```
DISP_FLAME.S
8行       .xdef              disp_flame
          .xref              gline              * 挿入
          .xref              apage              * 挿入

53,93行 jsr  (a4)     →   pea.l    (a1)       * 変更
                           bsr      gline      * 挿入
                           addq.l   #4,sp      * 挿入

63行      IOCS _APAGE →   bsr      apage      * 変更

WINDOW.S
13行      window:
                    pea.l   (a0)                * 挿入
                    jsr     setcliprect(pc)     * 挿入
                    addq.l  #4,sp               * 挿入

LINE.S
21行      IOCS _LINE  →   pea.l    (a1)       * 変更
                           bst      apage      * 挿入
                           addq.l   #4,sp      * 挿入
```

図2の変更を加えたプログラムとマシン語プログラミングに掲載されたプログラムをアセンブルしたら，付録ディスクに収められているすべてのソースファイル（変更したものは除く）をアセンブルしたものとリンクしてMAGIC.Xを作成します。これだけでもかなり高速化され，SIONも軽くなります。　　　　（影山　裕昭）

質問にお答えします

日ごろ疑問に思っていること，どんなことでも結構です。どんどんお便りください。難問，奇問，編集室が総力を上げてお答えいたします。ただし，お寄せいただいているものの中には，マニュアルを読めばすぐに回答が得られるようなものも多々あります。最低限，マニュアルは熟読しておきましょう。質問はなるべく具体的に機種名，システム構成，必要なら図も入れてこと細かに書いてください。また，返信用切手同封の質問をよく受けますが，原則として，質問には本誌上でお答えすることになっていますのでご了承ください。なお，質問の内容について，直接問い合わせることもありますので，電話番号も明記してくださいね。

宛先：〒108 東京都港区高輪2-19-13
NS高輪ビル
ソフトバンク株式会社出版部
「Oh! X質問箱」係

FROM READERS TO THE EDITOR

じりじりと太陽が照りつけるような季節になりました。学生たちには待望の夏休みが目前に迫り，すでに勉強など手につかず長期休暇計画を立てていることでしょう。夏休み後半にどたばたしないため，無謀な計画はやめとこうね。

◆6月号のような初心者のための講座はよいと思います。X68000もコンピュータの素人さんが買うようなパソコンになってきている現在，このような講座で「コマンドシェルも使ってみるか」というような人が出てくるでしょうから。吉田氏の記事にありますが，私は「コマンドシェルを使いながらSX-WINDOWも使うタイプB」の人が多いと思います。あと，せっかくウィンドウで使えるエディタが出たのですから，次はウィンドウで動く開発ツールを作ってほしいですね。エラーレポート用のウィンドウなどがあるといいのではないでしょうか。もちろんタグジャンプつきで。　猿渡　哲治(19)福岡県

◆初心者への環境構成法の説明が6月号の特集だが，マニュアルに書かれていることの繰り返しでは，実のある内容とはいえないと思う。むしろ，マニュアルを読む初心者の助けになるような言葉の意味（環境という言葉の定義など）や，編集部などで使用しているマシン環境の具体例の解説を書いてほしかった。

鈴木　大洋(22)茨城県

◆Oh!Xを購読し始めて今年でちょうど1年。付録ディスクを解凍したさにコマンドを覚え，記事中の単語の意味がわからなくてマニュアルを読みまくったおかげで，いまではCやアセンブラも少しは理解できるようになりました。然るに6月号の特集は……。内容の大部分はマニュアルに載っていることです。
この1年間，貴誌を読んで“Do it yourself”という編集姿勢を感じていた私にとって，6月号の特集は少し過保護かな，と思えました。しかし，マニュアルを箱の中に眠らせたままのユーザーにも，なんとかもう一度マニュアルを引っ張り出させようという編集部の熱意もちゃんと伝わってきましたよ。

今井　聡(22)島根県

ハガキを見ると，理解しやすかった。もの足りない。もっと簡単にして。と，意見の分かれた特集でしたが，皆さんはどのタイプにあてはまりましたか？

◆商業学のレポートの調べものをしに図書館に行った。いろいろと本を見ているうちにびっくりするような発見をしてしまった。みんなはすでに知っていることかもしれないけど，いま使っているシャープペンシルは，シャープの創始者の発明品なのだ。また，昔シャープはベルトのバックルを作っていた。これでX68000グッズのシャーペンとベルトは理解できた。次はツタンカーメンとポルシェがなにか？　だ。

安藤　道子(18)宮崎県

む～ん，現在の社長がツタンカーメンのマークをつけたポルシェに乗っているんじゃないかな。

◆わざわざ太字で，しかも大きく「西条秀樹」と間違えるなんてさすがは西川さんだ。

木下　卓也(19)埼玉県

西条秀樹というのは西川氏の親戚の友達であって，あの西城秀樹とは別人なのです。なんていいわけが通用するわけないよね。ごめんなさい。

◆う～む，このハガキが7月号に間にあえばの話なのだが，今日（6月○○日）は年間モニタで知られる安井百合江嬢の誕生日なのだそうだ。彼女にはいろいろとお世話になっているので誌面を借りてお祝いさせてもらおう。

「Happy Birthday!!」

間にあわなかったらただのバカだな。

神野　修二(16)愛知県

残念でした。これから神野君のことは「ただのバカ」と呼んであげましょう（ウソ）。

◆DRIVE ONの安井百合江(16)さんは本当に16歳なんでしょうか。文章がしっかりしすぎていて，61歳の間違いじゃないかと思うほどです(スミマセン)。　岡村　泰雄(19)東京都

安井さんはめでたく17歳になったようです。それにしても61歳だなんてあんまりじゃないですか。

◆また電話を止められちゃったよ～。NTTは高すぎるっ。長距離なんかいいから60kmあたりまでの電話料金をもっと下げてほしいな。話変わって，メモリを6Mバイトにしたのはいいけど調子にのってシステムを6Mバイトの環境にあわせたら（キャッシュ，RAMディスクetc……）またメモリが足りなくなっちゃったよお。

相馬　信隆(25)神奈川県

大丈夫，まだ6Mバイトも増設可能じゃあないですか。それでも足りなくなっても僕は知りませんけど。

◆最近思ったことですがSTUDIO Xで受け答えをしている方，パワーダウンしてきたみたいですね。以前はこのページだけでも腹を抱えて笑わせていただき，隣り近所に迷惑をかけてたほどだったのですが。もっとウハウハするようなあぶないエピソードを期待しています。

深井　雄一(19)秋田県

ぬう，厳しい指摘にしばし反省。皆さんも何か面白いことがあったら（ジャンルは問わず）ハガキに書いてくださいね。

◆3月15日，ついに買いました。いやあ，思えば長い年月でした。あれは6年前新聞のチラシ（だったと思う）に入っていたあの広告。「国産で初の0.30の壁を破った！」というコピー。そのときからほとんど変化のないデザインなのにいまでもインパクトがある。ああ，前から見ると「デブ」なにの横から見ると日本一。ユーノスが何だ，シルビアが何だ，時代は4WDだ，アルシオーネのものだ！　華門さん，師匠と呼ばせ

◀谷口　有香　北海道
女性ながら，なかなかインパクトのある絵です。サインペンでイラストを描くのは避けて，なるべくペンに慣れるようにがんばりましょう。

◀佐田　匠　千葉県
システムIIの作品はちょっと苦しいでしょうね。個人的にはシステムIの「ロンパーズ」を移植してくれるとうれしいです。

てもらいます。　　　　片山　勝喜(23)島根県

そのまま島根の華門正人を目指すのもいいですが，横断歩道を横切ってUターンするような危ないことはしないようにね。

◆とある筋から「じえーたい」の食料をもらった。無地の缶にでかく「赤飯」と書いてあって，なかなかゴツイ作りである。使用法曰く「沸騰湯中で約25分間以上加熱すれば，通常3日間は喫食できるが食前に温めればさらによい」むう，質実（？）剛健。　　　田中　真一(20)愛知県

で，おいしかった？

◆RENDRV.Xありがとうございます。さっそく打ち込んで使用しています。いままではDRIVE.Xを何回も実行してやっと目的のドライブの配列にしていたのだが，あっという間にできてしまう。あっけないやら嬉しいやらでとにかく重宝しています。おかげでGCCの使い方（オプションスイッチ）も少しずつわかってきたし。でもこれくらいは自分で作るべきなんでしょうね。

大塚　京吾(22)岐阜県

ぜひ，GCCをマスターして，RENDRV.Xを超えるようなツールを作れるようにがんばりましょう。

◆6月号，上級者のための環境考のはみだしで，編集室の反応は泉さんにけんかを売っているとしか思えないのだが。泉さんて嫌われているんじゃない？　かわいそう。高見　創(20)東京都

そうです，いじめられっこの泉氏は，日々皆の原稿催促に脅えながら仕事をしているのです（本気にしないでね）。

◆X68000を選んだ理由は？　グラフィックやサウンドが凄いから。というのが3年前の僕の答えだった。特に65536色というのが（当時は）凄いと思った。当然ゲームにも綺麗な絵を期待していたし，初めのうちはアーケードの移植ソフトばっかりだったのでグラフィックには満足していた。しかし，いま考えてみるとX68000のオリジナル（特にアクション系）ソフトのグラフィックに関していえば，ズームを除きほぼ全滅ではないだろうか。ナイアスやソルフィースのレビューでプログラムの技術は評価されても見た目とかはダメというのが多い。確かにそう思う。むしろPC-Engineなどのほうにセンスのよいものがたくさんある。がんばれズーム。

佐々木　美武(18)富山県

これだけの色数を使えるのだから，アーケードゲームレベルのグラフィックを目指してがんばってもらいたいものです。

◆可能な限りの最適化とアルゴリズムの改良で，1クロックでも速いルーチンを組むことこそがアセンブラの面白さだと思っています。そういう意味でMAGICやラインルーチン，ソートなどのプログラムはとても好きです。特に「MAGIC」。どんどん高速化して，最終的にはポリゴン対応になるといいと思います。POLYSYSを超えてAMIGAやIBMの3Dソフトに迫るものになってほしいです。自分でもやってみますが。

手賀　寛(16)福井県

高速化もいいですが，直接マシンをいじくり回すことができるのもマシン語の面白さのひとつでしょう。

◀尾形　雅治　広島県
ぱっと見た感じ，イースのイラストかな？　と思ったらキャメルトライですか。へえ，あのゲームも結構ナンパなところがあるんですね。

◀吉田　里志　宮城県
もうすでに夏だなあ，と感じさせるイラストですね。今年の夏はあまり暑くならないといいな（8月生まれのくせに夏は嫌いなんです）。

◆「X-OVER NIGHT」について。特に前半で書かれていたことについてはうなずける。いかに機能が強化されても，手間がかかって無駄な労力を使うのを考えれば手仕事で済ますほうがいいと私も思う。いささか機械に頼りすぎている。ちょっと話は違うけど，昔，日本語処理環境について特集したときに某氏がタコなワープロより速いエディタのほうが使いやすい，というようなことをいっていたが実際そのとおりで私も多機能ワープロよりもエディタで済ますことが多かった。6月号の特集もふまえてもう一度考えてみるのもいいかもしれない。

菊池　重幸(18)千葉県

使える機能は多くても操作がネックになっているケースが多いですね。多機能で，しかもエレガントな操作系を持ったワープロが早く発売になってほしいものです。

◆こんにちは。6月号の「愛読者アンケート」の結果発表は実によかった。それは，8ビット機をいまでも持っていて使っている人がいるということです。現在はX68000を中心に使っていますがたまにX1Cを使って遊んでいる人間として，いまだに自分と同じような人間がいるというのが実にうれしいことです。いつまでも現在使われている8ビット機に長生きしてもらいたいものです。　　　　石政　好康(20)富山県

時代に流されず，いつまでも大切に使っていきたいですね。

◆「夜明け」を入力して（なかなか気にいってます），OPMファイルに変換して聞いています。自分でも何曲か作ったのですが音色はプリセットのままなので，今度OPMD用にグレードアップしたいと考えています。しかし，曲がマイナーなので投稿するのはちょっと無理かな？　ところで昔のシンセなら勝るとも劣らないX68000のサウンド機能！　でも，もっと凄くなってほしいのは自分だけでしょうか。

石橋　誠(23)大分県

マイナーだろうといいものはいいのですから，遠慮せずにガンガン投稿してくださいよ。期待して待ってます。

◆SX-WINDOWのバージョンアップの知らせが待ち遠しい。早くお願いしたいものだ。「吾輩はX68000である」は罪な記事である。私を再びマシン語の世界に誘おうとしている。「X68000マシン語プログラミング」をもう一度，第1回から読んでいる自分が怖い。私は仕事はあるけど時間はない41歳である。　井門　清(41)愛媛県

やっぱり人生はマシン語ですよ。マシン語，マシン語……（悪魔の囁き）。

◆System-7CはMZ-700で動くことが凄いというより，MZ-700で動かそうとしたことのほうが凄いと思う。MZであそこまでできるのだからX68000ならもっと凄いことができる……はずなのだが。　　　　坂井　純一(21)茨城県

「その気になればなんでもできる」ということを，言葉だけでなく実際に行動してしまうのが古籏氏の凄いところですね。

◆この間，友人Kの家で大戦略をやっていたらKの祖父（70歳以上）が出てきて私と対決した。そしたらなんと8ターンでKの祖父が勝ってしまった。さすが戦争経験者と思う今日この頃であります。お年寄りを大切に。う〜んいいこというなあ。　　　　北原　恒(17)長野県

こらこら，冗談もほどほどに。

◆先日，やっとAMIGAが届いて，も〜うれしくてうれしくて。マニュアルすべて英語ってのが頭痛いかな。もちろんX68000は忘れてませんよ。なんせ，初めてパソコンらしいものを触った思い出の品。さて，ひとつ聞きたいんですけど私ぐらいの女性の方ってこの本を読んでいるんでしょうか。　　　　金本　朱美(20)三重県

あんまり浮気をしているとX68000がすねちゃうんで気をつけようね。

◆（善）のゲームミュージックでバビンチョのコーナーで「50分で2,000円は安い」とか「1,800円と安いよ」という言葉がありますが，もっと安くて長くてよいBGMがあります。それはキングレコードの「コナミ・ファミコンミュージックメモリアルベストVol.3」です。収録時間は74分で価格が1,500円。内容はファミコンの音とは思えないほどの素晴らしさ。特にニコルはお気に入り。もうひとつあって，同じくキングレコードの「ソリッドスネーク・メタルギア2」で

す。収録時間60分で1,500円。このゲームはMSX2用ROMカートリッジに音源（SCC出力ポート５）を載っけてて本体のPSG３音を使い素晴らしい曲だらけです。オープニングから「オオッ」という感じ。ぜひ聞いてみてください。

多田　哲也(20)兵庫県

（多）のゲームミュージックでモゲロンパのコーナーでした。

◆「響子inCGわ～るど」面白そうですね。専門的になるとわけのわからない言葉（文）が多くてCGって頭が悪いとできない，と思ってたのですが記事が女性らしく柔らかい感じの連載ということなのでこれからも期待しています。来年のベストライターは寺尾響子だ！

対馬　健治(25)青森県

ファンクラブ結成は，Oh!X編集室の許可が必要ですので注意してね。

◆僕はX68000ユーザーになって４年目を迎えるが僕の影響を受けて黒のX68000ACE，グレイのX68000EXPERT-HDが友人宅に導入された。さて次は誰に買わせようかな。と思いつつ，新たな洗脳作戦が始まるのであった。

太和田　誠(20)新潟県

そのまま新興宗教みたいになりそうで怖いなあ。

◆７年前くらいだろうか新聞かテレビかで面白いニュースを見た。「ソ連でひとり乗りの空飛ぶマシンが開発され数年後には実用化の見込み」白黒写真には軍人がスクーターのようなものに立って乗り，木の上まで浮かんでいるところが写っていた。なんだったんだろう。

中島　民哉(20)埼玉県

KGBの秘密兵器として本当に実用化されていたりして。

◆この間，夜にお墓の近くを通ったら“入れ歯”が置いてあった。うっすら白く輝くその姿は幽霊よりも怖かった。　金井　徳之(18)千葉県

やっぱり，お墓の住人がお供え物を食べるときに必要なのかなあ。う～ん，なんだったんでしょうかね。

◆今朝珍しく５時に目が覚めた。腹が減っていたのでコンビニへ弁当を買いにいったが売り切れだった。「しかたない」と思いながらなぜか「NEWカラムーチョ」と「辛みそソース味」のチップスを買い，一気に食べた。そうしたら胸焼けで半日ダウン。いったい早起きの意義ってなに？

米沢　明彦(23)東京都

「三文の得」どころか「半日の損」とは……。むちゃはしないようにね。

◆シグナトリーでマウスカーソルが国旗に変わったらそれを主人公の頭の上へ持ってきてこういいましょう。「ハタ坊だジョー」これでロード時間が多少つぶせます。清水　真也(18)千葉県

その技は哭きの竜でも可能です。マウスカーソルを相手の鼻に持っていき「ホジホジ」してやりましょう。

◆部屋にゴキブリが出た。まだ若いやつだった。たったそれだけのことなのにとっても気になる。今夜は掃除だ。ゴキブリを発見するまで。いまさらこんなことで悩むハメになろうとは，ゴキブリなんか嫌いだ。　大谷　篤志(18)福井県

関係ないけど，この間，道を走っているドブネズミを見てなぜか「かっこいい」と思ってしまいました。

◆本州ではもうすぐいや～な「梅雨」ですね。梅雨知らずの北海道では，じとじとな毎日がない代わりによく冷え込む日があります。元々本州生まれの本州育ちの私にとっては「寒い」ということは大いなる悪なのであります。その割には風邪をひきませんが。まあ，あとひと月もすれば真夏の太陽ギラギラですからそれまで耐えてくださいね。　宮本　明人(21)北海道

へいへい，耐えればいいんでしょう耐えれば（すねてどうする）。

◆近頃いちだんと目が悪くなってしまいました。目が悪いというのは本当に不便です。だけどよい点もあります。それは他人の視線が気にならないことと女の人が皆，綺麗に思えることでしょう。こういってしまうとなんですが，要するに顔がよく見えないということなんです。ということから私は女の人がより綺麗に見えるのは「夜目，遠目，近眼，笠の内」っていうのです。

藤原　彰人(21)岡山県

しょせんは自分を誤魔化しているにすぎないと思うんですけど。綺麗なものは綺麗に見えたほうがいいと思いますよ。

▲尾澤　宏　兵庫県
蛍光色に夜光色ですか。真夜中にうっすら光るX68000はちょっと怖い。でも，シースルー（透明）なら意外とかっこいいかもしれない。

▲板垣　央　千葉県
このイラスト，シンプルだけどなんか気にいっちゃたんですよね。今度は背景の白さがちょっと不満ですけど。背景のレイアウトもがんばろう。

◆最近某誌で読んだのですが，ディスプレイはブラウン管にかかっている高電圧が空中のほこりや重イオンを加速し，それが顔に当るとシミや皮膚ガンの原因になるんだそうです。女性スタッフの方はご注意あれ。平　直樹(18)東京都

そうなのか。でも編集部に女性スタッフなんていましたっけ？　あっ，E.O.さんは女性だったかもしれない。いてっ（殴られた）。

◆最近，学校に扇風機が入った。１クラスに２台ずつなのだが，あいにく風が当たらないところがある。僕の席はちょうどそこなのである。ちくしょう，席替えをしろ～。ちなみにうちの学校は新設校です。　藤田　祐一郎(17)熊本県

エアコンではなく扇風機というのが泣かせるねえ。

◆就職活動でソフトハウス会社を回っているけどコンピュータのことがいろいろ聞けて面白いです。コンピュータに興味がある人は，たとえソフト会社への就職を考えていなくてもセミナーだけでも参加してみることを勧めます。業界の裏話なんかも聞けますよ。

福田　秀昭(22)千葉県

しかし，必ずしも真実とは限らないので注意しましょう。

◆私は今年，某国立大学の情報工学科に入学しました。私は情報工学に入るやつらは，一応コンピュータについてなにか興味があるものだと思っていました。しかし，話をしてみるとおよそ８割の人が何も知らない＆興味がない，らしいのです。どうして情報に入ったの？　と聞くとほとんどの人が「偏差値が高いから」「就職がいいから」と答えてくれました。最近，こういうやつらと一緒に専門の授業を受けるのが嫌になってきました。僕が入ったのは地方大学ですが，ほかのところではどうなんでしょうか。

井上　聡(18)三重県

逆にいえば，井上君はきちんと目的意識を持って希望の学科にいけたのですから，他人のことなど気にしないで学生生活を楽しみましょう。

◆僕は医学系の学校に通っている者で，今度X1を使って心臓のシミュレーションを行おうと思っています。このために，いまアセンブラを使ってプログラムを組むために懸命にアセンブラの勉強をしています。予定として画面自体は（８色しかないため）白，黒，水色と動・静脈に赤，青を割り当てるつもりです。できれば心臓のみではなく，心臓・肺・腎臓・肝臓などのシミュレータも作りたいと思っています。

岸部　淳一(20)北海道

「遥かなるオーガスタ」のPOLYSYSみたいなものを作って，「ミクロのゴルフツアー」なんてゲームができたりして。

◆今年，大学４年である私には卒業研究という最後の授業がひかえている。そこで研究に必要だということで，研究室にX68000XVIを導入させようと計画している次第である。PC-9801は捨てるほどあるのでX68000でしかできない研究

とは何か？　と考えるこの頃である。
長崎　洋(22)神奈川県
現実に導入に成功した人間もいますからがんばりましょうね。

◆とうとう私も受験生になりました。ところで新声社のオリジナルTシャツ（6月号のプレゼント）真ん中の人は代々木ライブラリ「前田の物理」のキャラクターそっくりだ。
安保　和幸(17)愛知県
「前田の物理」なんて聞くと暗い浪人時代を思い浮かべてしまうけど本当にそっくりですね。

◆「風の谷のナウシカワイドコミック第5巻」が出たのでもちろん買いました。ストーリーを忘れていたので1～4巻を読み直した。そして，ついに5巻を読んだ。面白い。読み終えたときはなんとAM4：30，あー時間が過ぎるのは早い。しかし，6巻を早く読みたい。
三沢　弘之(19)神奈川県
風の谷のナウシカも連載開始してからそろそろ10年がたちますね（中断が長いおかげで)。はたしてこの物語に終わりがあるのでしょうか。

◆4月の情報処理試験1種，自己採点では8割程度いったのでひと安心。でも当日は欠席者が目立ちましたね。約半分ってとこでしょうか。自信があろうがなかろうが，まず受けないことには始まらないのに。私なぞ直前に3時間勉強したきり。それでも受けにいく神経の図太さ……。ああ，受かっているといいな。
西田　文彦(20)神奈川県
世の中そんなに甘くないぞ。しかし，なんとかなってしまうときもありますけどね。で，結果はいかに？

◆さすがメタボールは凄い。私はサイクロンユーザーなのでうらやましいかぎりです。しかし，このところアンスの広告がないのでそろそろ「サイクロン・メタボール」が出るのでは？　早くこいこいメタボール。もっとよくなれメタモデラー。おっと，その前に拡張I/O BOXとメモリを……。
溝江　純一(21)奈良県
さて，出費はどれくらいになるんだろう。あまり考えたくありませんね。

◆最近，SX-WINDOWにどっぷりの毎日です。Ver1.0だから遅い，使いづらい，という問題がありますがノートウィンドウをガバガバ開いて，有機的に文章を作る便利さを知って以来，滅多にワープロを使うことがなくなりました。でも，エディタのほうが使いやすそうなので早くVer1.1が発売されないかな。また，PICファイルもSX-WINDOW上で見ることができるので画像ファイルの整理も楽ですね。
田浦　達也(30)千葉県
すでにVer1.1も発売されました。さっそく購入されて使っていることと思いますが使い心地は期待どおりでしたか？

▲杉本　秀昭　宮城県
猫耳ならず鳥の羽耳少女（？）ですね。全体的にきれいに描けてますけど，仕上げが少し粗いかな。縮小したら気にならないだろうけど注意してね。

ぼくらの掲示板

●掲載ご希望の方は，官製ハガキに項目(売る・買う・氏名・年齢・連絡方法……)を明記してお申し込みください。
●ソフトの売買，交換については，いっさい掲載できません。
●取り引きについては当編集部では責任を負いかねます。
●応募者多数の場合，掲載できない場合もあります。
●紹介を希望されるサークルは必ず会誌の見本を送ってください。

仲間

★「OREGA」。当クラブでは，年間10回程度の会報発行を中心に幅広く活動しています。会報はプログラミング講座（「はじめての3D」連載)，ハードウェア講座，テクニカル情報，ゲーム(「男のアミューズメント道」連載)，パソコン通信情報，読書案内，エッセイ（「OLからひ・と・こ・と」連載)，SF，イラストなど盛りだくさんです。
入会希望の方は入会案内を送りますので124円分の切手を同封のうえ，郵便番号，住所，氏名を明記して下記までお送りください。〒910　福井県福井市文京4-9-5メゾン山本201　新海敏之・「入会希望X」係

★このほど，我が「PC-AI」では磁性面のパワーアップのため，協力的な会員を募集します。購読のみの方はご遠慮願います。入会希望の方は62円切手を同封のうえ，資料請求してください。折り返し案内と申込書，会費払込の振替用紙を返送します。〒735　広島県安芸郡府中町大須3-5-6　丹下　優(20)

売ります

★CZ-800C用G-RAM，CZ-8GRを3,000円で。デジタル分配器，KSW-D（DISMATCH）を2,000円で。共に箱，説明書つき。連絡はハガキか往復ハガキで。〒240-01　神奈川県横須賀市秋谷1-5-34　榎本　秀俊(23)

★1Mバイト増設RAMボード（CZ-600C専用）新品を送料込みで15,000円で。連絡は往復ハガキでお願いします。〒110　東京都台東区谷中6-1-13　荒居　稔宣

★ハードディスク「CZ-620H」を送料込みで15,000～20,000円で。箱，マニュアルはありません。連絡は往復ハガキでお願いします。〒790　愛媛県松山市平和通2丁目1-7エスペランス405号　浅野　公昭

★カラーイメージユニット「CZ-6VT1」を送料込みで30,000円で。ケーブル，ターミナルBOX，マニュアルあり。箱はなし。包装は完璧にやります。連絡は往復ハガキでお願いします。〒940　新潟県長岡市城内町3-5-10　渡辺　良太郎

★カラーディスプレイ「CZ-850DR」を20,000円，データレコーダ「CZ-8RL1」を6,000円で。共に送料込みです。箱，マニュアル，ケーブルあり。外観も良好です。連絡は往復ハガキでお願いします。〒361　埼玉県行田市持田5-13-2　吉本　邦雄

★ビデオボード「CZ-6BV1」を15,000円（送料込み）くらいで売ります。箱，マニュアル，付属品あり。連絡は往復ハガキでお願いします。〒381-02　長野県上高井郡小布施町851　鈴木　理星

★「CZ-6PV1」（カラービデオプリンタ）を65,000円以上で。高く買ってくれる人優先します。完動美品，付属品，箱，すべてあり。プリントシートがもう2枚しかありません。詳細および連絡は往復ハガキでお願いします。〒230　神奈川県横浜市鶴見区鶴見中央3-20-9-1103　金田　宏卓(18)

買います

★MZ-711/721の「プロッタプリンタ設置場所のカバー」を731円くらいで（ネジをつけてくださる場合ならば800円くらいで)。送料当方負担，連絡は往復ハガキでお願いします。〒227　神奈川県横浜市緑区つつじヶ丘1-5 B-501　石田　伯仁(18)

バックナンバー

★Oh!X1989年10月号，1990年4,5,8,12号を各1,500円（送料込み）で買います。切り抜き不可，キズあり可。連絡は官製ハガキでお願いします。〒254　神奈川県平塚市田村5214-1田村団地147　石田　勝利(29)

編集室から from E・D・I・T・O・R

DRIVE ON

このコーナーでは，本誌年間モニタの方々のご意見を紹介しています。今月は6月号の内容に関するレポートです。このメンバーとも（一部の方を除き）今月でお別れです。1年間，本当にご苦労さまでした。

●何の予備知識もないままにX68000を買った人にとり，6月号の特集「初心者のための環境構成術」は十分に役立ったのではないかと思います。なにしろ，かなりわかりやすく書いてありましたから。しかし，「A>」とつきあいはじめてから1年以上もたっている僕にとっては，あまり役に立つといえる内容ではありませんでした。ましてや，SX-WINDOWもWP.Xももともと使う気がないため，興味深く読めたのは，泉氏の「上級者のための環境考」ぐらいでした。

泉氏の「FDDはA,Bに固定しよう」という提言はいい考えだと思います（実際，僕もそうしている)。しかし，これは下手をすると「FDDはA,B」というだけでなく，「A,BはFDD」という固定概念が生じかねません。ですから，かの提言には「なるべく」をつけ加えるべきでしょう。

高橋　毅(20)　X68000 PRO, MSX2　埼玉県

●「PC-9801用マウスをつなぐ」について。「肩透かし」をくらったような感じでした。でも，発想はすばらしいと思います。作者の方はマウスを何台も壊しておられるのでしょうか。私は残念ながらマウスを1台も壊したことがありませんから，この発想に「してやられた！」と思いましたね。

そうなんです。仮にX68000用のマウスを新品で買ってきて，この改造（？）を行ったとしても，そのことで，安いものや操作性に優れたものなど自分にあった1台を多数の製品の中から選択できるようになるわけですから，トータルではきっと経済的であると見なせるのですよね。

難易度はきわめて低いものである（というより，「アナログジョイステイックの製作」が難しすぎた）と思いますが，ひとつ難点をいえば，作り方の説明がものたりません。つまり，「基盤の指定された場所にコードをハンダ付けする」としか書いていないわけで，「なぜここに接続すれば動くようになるのか」が不明であるというわけです。

PC-9801のマウスをただ単にX68000につなぐだけならば，どうしてここにつなぐと動くようになるのかなどの理由は必要ありません。しかし，もし改造に失敗したとき，理屈がわかっているのとわかっていないのとでは，デバッグのやりやすさに雲泥の差が出ます。

また，単に改造の方法を記載したものだけでは，それ自身に資料としての価値はありません。さもしい根性かもしれませんが，これをヒントにマウスに内蔵されている富士通のICの仕様がわかるようにもなるし，もしわかれば，完全自作でPC-9801のマウスをX68000につなげる可能性も開けてきます（そのICが安く入手できれば，さらに経済性に拍車がかかるわけです)。

つまり，今月の記事のようなものであると，ほかへの応用がきかなくなってしまう，ということなのです。

汎用性を持たせると内容が難しくなってしまうことは，なるほど避けようのないことかもしれません。でも今月にかぎっていえば，「ハードウェア工作入門」に比べてもはるかに簡単な内容になるような気がします。無理な注文かもしれませんが，難易度と汎用性を両立した内容を今後も期待します。

浅野　憲(19)　X68000 PRO, X1turbo III, X1F, MZ-80C, PC-9801RL, PC-6001, FM-77L2, M5Jr., PC-1245　大阪府

●「System-7C」の解説記事ですが，内容的にはあまりハードウェアの操作には踏み込まず，一般的なシステム設計論となっているので，他機種ユーザーでも理解できる内容だったと思います。それより，いまのマイコンユーザーの中にこういう内容のわかる人がどのくらいの割合なのかのほうが心配です。あと，些細なことでもうしわけありませんが，90ページのMZ-700の仕様が間違っています。テキストは40桁×25行です（私は元MZ-700ユーザーです)。

泉　昭彦(21)　X1turbo, PC-E500　東京都

●「きょうこ」さんってのは，寺尾さんのことだったんですか。誰かなあとずーっと考えていたんですよ。いま，私も描いているものがあるのですが，少しアイデアが煮詰まってきているし，なかなかモデリングをする暇がないんです。それに友人がスキャナを借りていってしまったので，いちばん重要なところが浮かばない。まあ，それはさておき，このコーナーでは商業レベルのCG（早い話，プロのCGですね）の制作過程なども紹介してほしいなと思います。

話は変わって，先日FLOAT2.X ver.2がどれだけ速いのかという実験をしました。彩クロン expressで水平ラインピクセルを150（たぶん，そうだった）に設定，赤い球を描いてみたんです。球のアトリビュートは，いうまでもなく乱反射体。FLOAT2＋.X＆HSMUL.X（彩クロンについていた32ビット整数演算高速化プログラム）で28分くらい，FLOAT2.X ver.2（電脳倶楽部の6月号に入っていた）では20から22分くらいでした。当たり前のことなのかもしれませんが，新FLOAT2.Xを組み込むとHSMUL.Xは使えません（アドレスエラーが出る)。それでも，速いですねえ。さっそく，ほかのシステムのFLOATも差し換えました。正直いって，HSMUL.Xがあるし，FLOAT2＋.Xだからたいして差は出ないと思っていたのに。8分の差ですよ。すごいすごい。

安井　百合江(16)　X68000 PRO　愛知県

ごめんなさいのコーナー

7月号　LIVE in '91

P.149　「歩いていこう」のプログラムリスト の中に誤りがありました。

```
1260 goto 4000
      ↓
1260 goto 2400
```

と訂正してください。

6月号付録ディスク　APIC.FNC

「APIC.FNC」は正常に動かない場合があったようです。詳しくは今月の「Oh!X質問箱」の記事を参照してください。もうしわけありませんでした。

バグに関するお問い合わせは
☎03(5488)1311(直通)
月～金曜日16：00～18：00

お問い合わせは原則として，本誌のバグ情報のみに限らせていただきます。入力法，操作法などはマニュアルをよくお読みください。

また，よくアドベンチャーゲームの解答を求めるお電話をいただきますが，本誌ではいっさいお答えできません。ご了承ください。

今月のOh!X編集部は印刷工房だった

▼今月の特集は「印刷の世界へ」と題し，プリンタへの出力を中心にお送りしました。思いどおりの色に，あるいは思いどおりのかたちに印刷するということはむずかしいことです。しかし，それが達成されたときの楽しさは，目に見えるだけに格別でしょう。今回の特集のために編集部ではかなりの量の紙，インクを消費しました。関係ない人たちが見たら無駄だというかもしれませんが，カラーページを見た皆さんはそう思わないですよね。
▼また，今月は表紙でもおわかりのように，特集関連で「IO-735X-B」，「JX-220X」，MIDI機器の「SC-55」，そして3.5インチディスクドライブ「TS-3XRI」と多くの製品を紹介することができました。特に，ツクモの「TS-3XRI」は他機種とのデータのやりとりに便利ということで，切望されていた方も多いと思います。これを機に，真にユーザーが待ち望んでいる周辺機器が，シャープから，そして，さまざまなサードパーティから，続々と発売されることを期待したいですね。
▼「DRIVE ON」のコーナーでも書かせていただきましたが，6月号まで年間モニタを務めていただいた方，ご苦労さまでした。また，新しい年間モニタの方々には7月号とレポート用紙をお送りしました。さっそく，面白そうなレポートが続々と返ってきています。紹介は9月号ですので，お楽しみに。
▼Oh!Xでは，OPMDに代わる新しいX68000のミュージックドライバを開発中です。あわせてAD PCMによるサンプリング音の標準ライブラリやそれらを加工するツールなどを収めたディスク（2枚組かな）を今秋には発売する予定ですので，どうかご期待ください。
▼今月は「マシン語カクテル」，「DōGA・CGアニメーション講座」がお休みです。さらに，「シミュレーション入門」が今月も休載となってしまいました。作者の華門真人氏は大変多忙なようです。「来月こそは掲載を」とがんばっていますが，それ以降は不定期な連載となりそうです。

投稿応募要領

●原稿には，住所・氏名・年齢・職業・連絡先電話番号・機種・使用言語・必要な周辺機器・マイコン歴を明記してください。
●プログラムを投稿される方は，詳しい内容の説明，利用法，できればフローチャート，変数表，メモリマップ（マシン語の場合）に，参考文献を明記し，プログラムをセーブしたテープ（ディスケット）を添えてお送りください。また，掲載にあたっては，編集上の都合により加筆修正させていただくことがありますのでご了承ください。
●ハードの製作などを投稿される方は，詳しい内容の説明のほかに回路図，部品表，できれば実体配線図も添えてください。編集室で検討のうえ，製作したハードが必要な場合はご連絡いたします。
●投稿者のモラルとして，他誌との二重投稿，他機種用プログラムを単に移植したものは固くお断りいたします。

あて先

〒108 東京都港区高輪2-19-13 NS高輪ビル
ソフトバンク出版部
Oh!X「㋢㋮㊔」係

SHIFT・BREAK

▶楽あれば苦あり，ということわざがあるが，それはどうやら私にはあてはまらないらしい。なにしろ必要な時間を合計し，期限までの時間と比べると圧倒的に前者が勝つ。不幸を嘆くひますらなく，レポートに追われる私に明日はあるのだろうか……？というわけで，編集後記は不幸な話がウケるって聞いたのですが，本当なのかしらん？（八）
▶今月号が発売される頃，世間はもうすっかり「夏っ！」なんでしょうね。今年は冷夏らしいですが，夏といえばやっぱり日焼けです。いまでこそ真っ白な私ですが，中高時代は水泳部員で必要以上に真っ黒でした。ただ困ったのは競泳パンツの模様がそのまま下にもうっすら焼けてしまって，銭湯でも必要以上に目立ってしまうことでしたとさ(笑)。（哲）
▶パソコンのプリンタ用紙って，なんであのサイズなんだろう。あのミシン目がついて端に穴の開いているヤツ。ワープロ代わりに使っている私はA4とかB5の紙に印刷してくれるほうがうれしいんだけど，プリンタもワープロもそういう紙への対応が甘い。プリンタ用紙ではあとでコピーがかけづらいし，なにより紙がもったいないじゃん，ねえ。（浦）
▶へっへっへ。ついにラップトップパソコンを買ってしまった。PC-386noteW。ジャストA4サイズも魅力だし，本当に持ち運べる重さもよい。紙袋に入れて持ち運んでいるので，周りの人は誰もコンピュータとは気づかない。図書館でおもむろに取り出したときに浴びる視線はたまらないものがある。ラップトップでも実力派だよ，こいつは。（S.K.）
▶毎週月曜日フジテレビ系列で27：30（火曜日AM03：30）からやっている「13日の金曜日」は面白い。電動ノコギリのジェイソンとはまったく無縁。骨董品屋の前の店主が売りさばいてしまった呪われた悪魔の品々を，3人組の新しい店員が毎週取り戻していく……，というなんとも奇妙なストーリーの1時間ドラマです。（善）
▶まして，夏である。昼間は暑くて仕事にならないので，明け方に原稿を書く。朝5時になると，ふらふらと近所の川を中心としたなかなか広い公園に散歩にいく。時々，珍しい鳥がいてうれしい。先日は10年振りにゴイサギに出会った。朝5時は老人の時間である。散歩老人や体操老人やジョギング老人の大バーゲン。それは不思議な光景だ。（K）
▶「プロゴルファー猿」だけでしかゴルフを知らないが，「遥かなるオーガスタ」を買った。ただボールを打って穴に入れるだけかと思っていたが，ちゃんと風速やクラブの飛距離を考える必要があったとは（当たり前だね）。ゲームとはいえ奥が深い。初めてプレイした成績は9ホールで60オーバー（笑）。でも，これから当分は遊べそうだ。（KO）
▶いきなりBJ-10Vとハードディスクを衝動買いしてしまった。もちろん支払いは，いつもにこにこ現金一括払いだ。ハードディスクは，予約だけでまだ届いていないけど日曜日にはくるようなのでとっても楽しみ。桁の少なくなった貯金通帳なんか関係ないや。貧乏だっていいじゃないか。さあ，がんばって仕事をバンバンやるぞー。（J）
▶朝5時は老人の時間らしい。しかし，うちの近所の公園では朝10時ごろ，老人のゲートボールを見ることができる。そして，その近くでは大勢の子供たちが水浴びをしている。まさに人生の縮図を見るような通勤風景なのだ。でも，なんでゲートボールをやっている老人同士はあんなに仲が悪いんだろうか。たぶん「燃えるスポーツ」なんだろうな。（A）
▶我が家にSC-55がやってきた。いまは忙しくてなかなかかまってあげられないけど，もちっとしたら遊ぶんだい！　しかし我が家は楽器関係が多いこと多いこと。ギター4本にデカいアンプが2台＆持ち歩き用1台，シンセサイザにエレピにドラムマシン，MTR，各種エフェクタ……。このうえもう1台シンセを買う予定。ふう，どこで寝ようかな。（E.O.）
▶朝5時は……情けないのでやめとこう。「パロディウスだ！」は2周目で真の姿を現す。動いていただけで凄いと思ったのは甘かった。1周目の倍の処理を軽々とこなす。なぜこんなモノが作れるんだ？さて，特集のサンプルで作ったTシャツを若干名にプレゼント（M寸のみ）。例によって0番で応募のこと。（マツダのルマン制覇がちょっとうれしいU）
▶パルコの抽選会ではパソコン(FM TOWNS)が一役買っていた。好きなパネルをマウスで選んで裏返す簡単なものだが，客のほうはタッチパネルのつもりか直接画面を指差し，係のおねえさんがマウスを操作するという始末。ハズレると「そこじゃない。もう一度やらせろ」と怒る客も。一般人ってそういうものなんだなと改めて実感させられた。（T）

microOdyssey

言葉で表現することの難しさ，コミュニケーションの難しさ。これが入社して3カ月たった僕がいちばん感じたことだ。どのようにすれば多くの人に理解してもらえるか，誤解をまねかずに自分の意志を伝えられるのだろうか。人それぞれ自分なりの考え方というものを持っているのだから，賛成意見があれば反対意見があるのは当たり前のこと。

賛同者だけを集めて自分に合わない意見をつっぱねることは簡単だし，そのほうが気持ちがいい。しかし，意見のぶつかりあいは必然的に発生する。それによって相手を理解していくわけだが，自分の中に相手の意見を受け入れることは本当につらい。納得のいくまで議論を進めずに投げ出し，自分のいいたいことが伝わらずいらついたことは誰しも経験しているだろう。

ま，すべてが自分と同じになればこんなことに悩まされずにうまくいく。皆が同じ思考をしていれば，衝突もなくなる。なにげなく書いてみると確かに楽で気持ちがいい世界だし，自分たちが気づかなければひとつの理想郷ともいえるかもしれない。だが，たいていの人はこんな世界は願い下げだろう。自分が他人と同じだなんて考えただけでも気持ち悪い，と。

さて，このようなことで悩むようになったのは，「STUDIO X」のコーナーを担当することになってからだ。読者とのコミュニケーションをとるいちばんの窓口であり，毎月送り返されてきた愛読者ハガキの中からOh!Xの看板を背負った僕がハガキを選択していく，結構いろいろな意見，話が聞けて楽しい。そしてこのハガキはなにを求めているのか？　なにを主張しているのか？　と，短い文面から読み取っていく作業は難しいけれど楽しい仕事だ。

で，問題となるのがそれらの意見，主張をどういった形で雑誌に取り入れるかだ。読者の意見を無視して自分の意見を押しつけるようなおこがましいことができるわけもなく，仮にやってしまったら主義に合わない読者はすぐに離れてしまうのは明白だ。これはゲームレビューだろうとリストの解説だろうと同じこと。常に読み手のことを考え文章を構成していかなくてはならない。ひととおりの解説をつけただけの文は無視され，趣味に走りすぎると同意を得られない人たちからのブーイングがある。言葉は悪いが，気に入らないことがあるとめざとく見つけてくる敏感な読者をいかにしてだまくらかし，どうやったらそれとは気づかせずにこちらのペースに巻き込むことができるのだろうか。

入社する以前に，ライターとしてこの雑誌に関わってきたから，どのように対処したらいいかわかっているはずじゃないのか？　と読者の皆さんは思うかもしれない。しかし，まだまだ経験が足りず思ったことを行動に表わせないでいる。受け手から送り手側に立場が逆転したことでとまどい，しかも気ばかりあせって妙に落ち着かない。とりあえず行動に移す，そうしようとすると，本当にそのままでいいのか？　という思いが浮かび上がって躊躇してしまう。すべての人に理解してもらうことができなくても，より多くの人たちに賛同してもらえるようになりたい。これは欲張りだろうか。

とりあえず，当たり前のことが当たり前のようにできるようにがんばりたい。　（J）

1991年9月号8月19日(月)発売

特集 Brush up your MAGIC.
・MAGICの高速化と周辺整備
特別企画 Oh!Xの正しい読み方
X1用アスレティックアクションゲーム
詳報 サウンドキャンバスSC-55
新製品紹介
Mu-I Super，マルチワープロPRO-68K

バックナンバー常備店

東京	神保町	三省堂神田本店5F 03(3233)3312
	〃	書泉ブックマートBI 03(3294)0011
	〃	書泉グランデ5F 03(3295)0011
	秋葉原	T-ZONE 7Fブックゾーン 03(3257)2660
	八重洲	八重洲ブックセンター3F 03(3281)1811
	新宿	紀伊国屋書店本店 03(3354)0131
	高田馬場	未来堂書店 03(3200)9185
	渋谷	大盛堂書店 03(3463)0511
	池袋	リブロ池袋店 03(3981)0111
	〃	西武百貨店9F コンピュータ・フォーラム 03(3981)0111
神奈川	横浜	有隣堂横浜駅西口店 045(311)6265
	〃	有隣堂ルミネ店 045(453)0811
	藤沢	有隣堂藤沢店 0466(26)1411
神奈川	厚木	有隣堂厚木店 0462(23)4111
	平塚	文教堂四の宮店 0463(54)2880
千葉	柏	新星堂カルチェ5 0471(64)8551
	船橋	リブロ船橋店 0474(25)0111
	〃	芳林堂書店津田沼店 0474(78)3737
	千葉	多田屋千葉セントラルプラザ店 0472(24)1333
埼玉	川越	黒田書店 0492(25)3138
	川口	岩渕書店 0482(52)2190
茨城	水戸	川又書店駅前店 0292(31)0102
大阪	北区	旭屋書店本店 06(313)1191
	都島区	駸々堂京橋店 06(353)2413
京都	中京区	オーム社書店 075(221)0280
愛知	名古屋	三省堂名古屋店 052(562)0077
	〃	パソコンΣ上前津店 052(251)8334
	刈谷	三洋堂書店刈谷店 0566(24)1134
長野	飯田	平安堂飯田店 0265(24)4545
北海道	室蘭	室蘭工業大学生協 0143(44)6060

定期購読のお知らせ

Oh!Xの定期購読をご希望の方は綴じ込みの振替用紙の「申込書」欄にある「新規」「継続」のいずれかに○をつけ，必要事項を明記のうえ，郵便局で購読料をお振り込みください。その際渡される半券は領収書になっていますので，大切に保管してください。なお，すでに定期購読をご利用の方には期限終了の少し前にご通知いたします。継続希望の方は，上記と同じ要領でお申し込みください。

海外送付ご希望の方へ

本誌の海外発送代理店，日本IPS（株）にお申し込みください。なお，購読料金は郵送方法，地域によって異なりますので，下記宛必ずお問い合わせください。

日本IPS株式会社
〒101 東京都千代田区飯田橋3-11-6
☎03(3238)0700

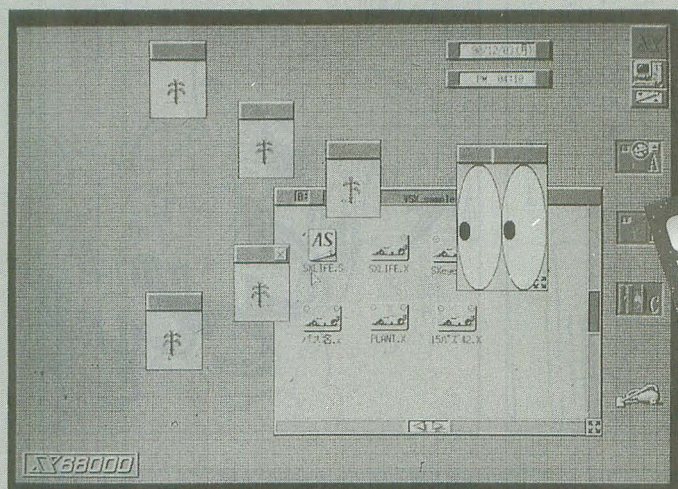
Oh!X 8月号
■1991年8月1日発行　定価600円（本体583円）
■発行人　孫　正義
■編集人　橋本五郎
■発売元　ソフトバンク株式会社
■出版事業部　〒108 東京都港区高輪2-19-13　NS高輪ビル
Oh!X編集部　☎03(5488)1309
出版営業部　☎03(5488)1360 FAX 03(5488)1364
広告センター　☎03(3297)0181
■印　刷　凸版印刷株式会社

待望出来!! この本で始まる SX-WINDOW時代

SX-WINDOW プログラミング

吉沢正敏●著

●B5変型判・468ページ●定価4,500円

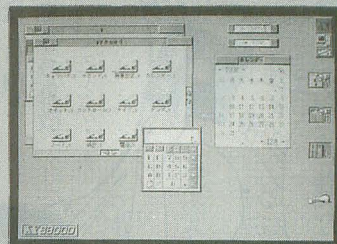
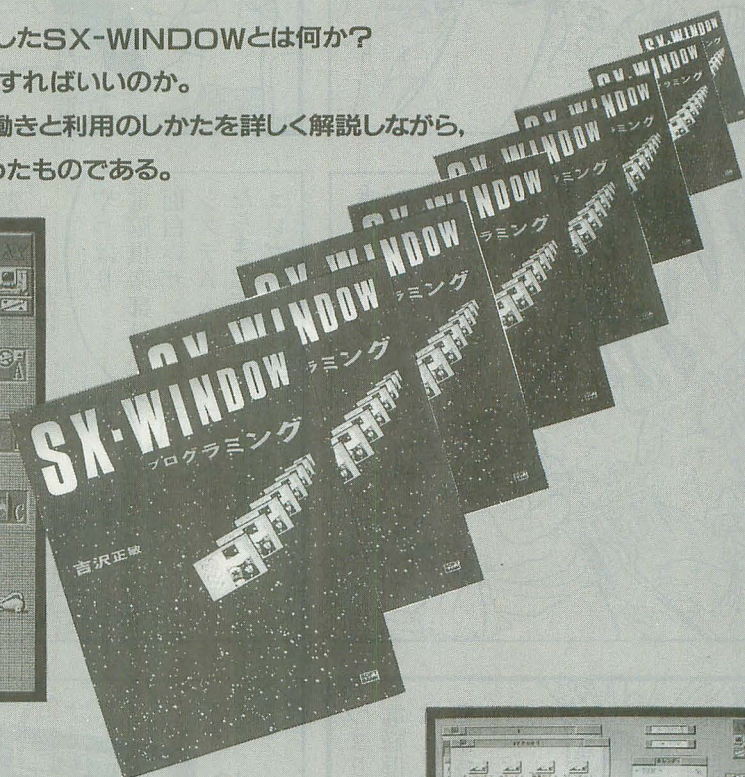
X68000にマルチタスク・マルチウィンドウ環境をもたらしたSX-WINDOWとは何か?
このSX-WINDOW上でプログラミングするにはどうすればいいのか。
本書は，SX-WINDOWを構成する各マネージャの働きと利用のしかたを詳しく解説しながら，SX-WINDOW上でのプログラミングの実際をまとめたものである。

本書のおもな内容

第1章 SX-WINDOWとは何か

第2章 各マネージャの働きと利用方法

第3章 プログラミングの実際

第4章 SXコール・リファレンス

APPENDIX SX-WINDOWのための用語集ほか

好評既刊

X68000 マシン語プログラミング 入門編

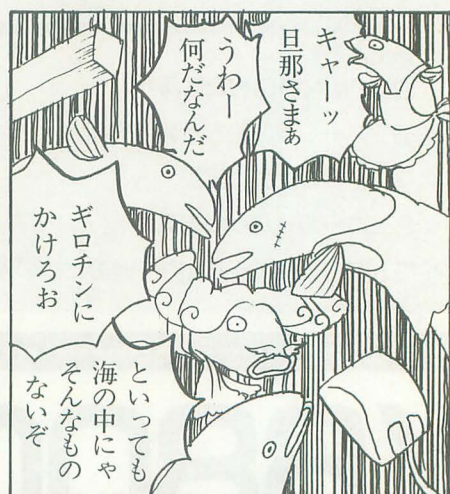
著●村田敏幸

●B5変型判・388ページ●定価2,800円

マシン語に限らず，プログラミングに関する知識/技術は，実際にプログラミングする中でこそ身につき，磨かれるものだという不変の真理にもとづいて書かれた"実践的マシン語プログラミングの書"である。実際に自分の頭と体を使って読み進んでほしい。巻末の用語集も好評である。

●発売元 ソフトバンク株式会社出版事業部 〒108 東京都港区高輪2-19-13 NS高輪ビル TEL03(5488)1360

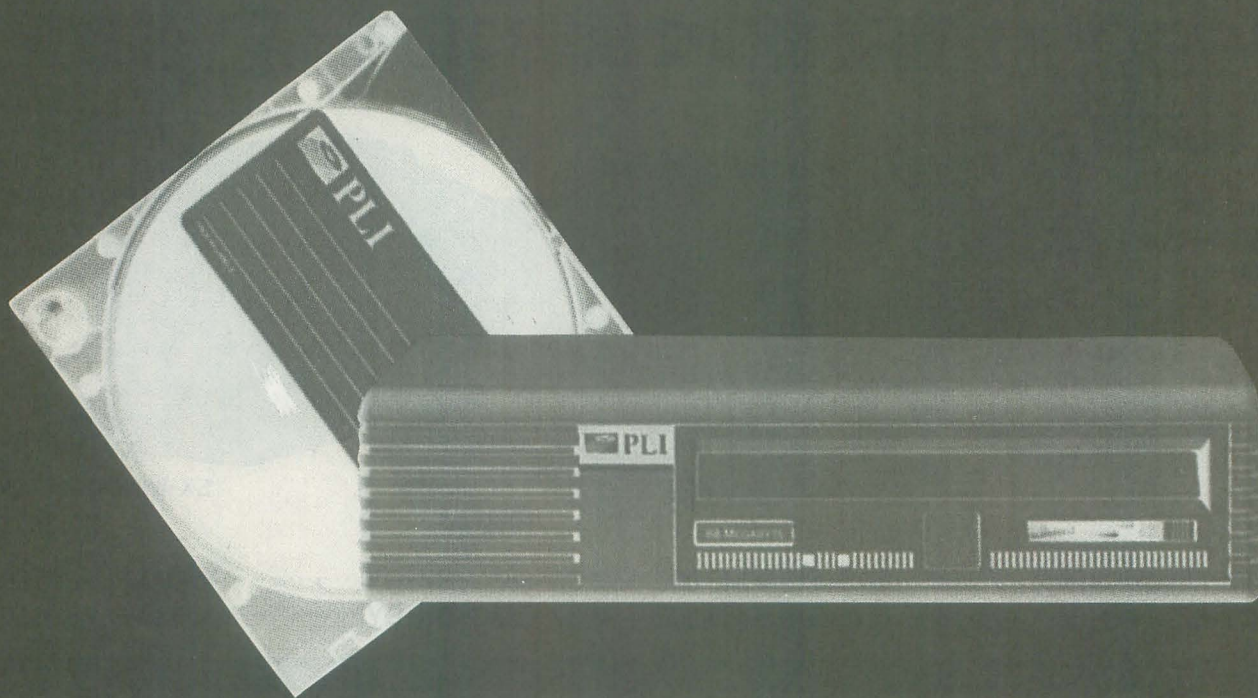
満開の電子ちゃん

作・え　岡村祭

「読めば読むほど前の号が欲しくなる雑誌」というものがありますが、「電脳倶楽部」は、まさにそれだと思います。便利なプログラム、ビープ音、グラフィック、PDD等、豊富な内容。「しまった、もっと早く購読しておけば良かった」と思うこと請け合いです。そんな時も、品切れなくバックナンバーを購入できるところが嬉しいところです。また、掲載された殆どのプログラムのソースリストが公開されているのも、大変ありがたいです。パソコン通信と違って、電話代を気にせずに済むところも嬉しいですね。

米山一輝（大阪府）

THE Removable HDD

メディア交換可能な新世代ハードディスク

PLI Infinity 40 turbo

リムーバブルハードディスクはフロッピーディスクや光磁気ディスクと同じようにメディアを抜き差しできる新世代のハードディスクです。
交換が可能になることによりデータのバックアップなどをメディアごと行なうことも可能ですし、他のInfinity 40 turboユーザーとのデータ交換なども簡単に出来るようになります。
PLI社のInfinity 40 turboはアメリカでマッキントッシュやIBM PC用として既に多大な評価を受けておりその品質は万全です。気になる平均シークタイムも20msecと固定型のハードディスクに引けを取らない高速です。BASIC HOUSEはこの大変便利なデバイスをぜひX68000ユーザーの皆さんにも使用していただきたいと思い、販売を開始いたします。

★平均シークタイム20msce
★メディア1枚当たりの容量40Mバイト
★連続耐久時間30000時間以上
★SCSIインターフェース対応
★X68000用SCSIケーブル、ターミネータ付属
★メディア1枚(40Mバイト)の価格はわずか¥18,000

BASIC HOUSE 計測技研
お申し込み・お問い合わせは 0286-22-9811(代)
〒321 栃木県宇都宮市竹林町503-1 FAX0286-25-3970

ハードディスクを内蔵させた

超高速 12msec

XVI(エクシヴィー)がおいしい

夏休み特別価格

8月号のみ! 従来価格よりさらに**¥20,000**-OFF!!

※お申し込みの際には必ず「Oh! X8月号の広告のXVI」と書き添えてください。

40M バイト内蔵モデル
XVI40
従来価格 ¥378,000
BH特価 **¥358,000**

100M バイト内蔵モデル
XVI100
従来価格 ¥428,000
BH特価 **¥408,000**

200M バイト内蔵モデル
XVI200
従来価格 ¥528,000
BH特価 **¥508,000**

通信販売のみ!一般販売店では扱っておりません。

※表示価格はハードディスクを内蔵させた本体のみの価格です。
※ディスプレイなどは別にお求め下さい。
※使用しているドライブの関係上、立ちあげには電源投入後約15秒で一度リセットをする必要があります。

最大メモリ8Mバイト

KGB-X68PRKⅡ

新発売!

- 8M増設メモリと数値演算プロセッサが1枚のボードに収まります。
- 従来品(KGB-X68PRK)に比べて大幅なコストダウン。
- メモリ容量2M/4M/6M/8Mの4種類、それぞれに数値演算プロセッサ有無のモデルを用意しました。
- 数値演算プロセッサ無しモデルではMC68881RC16の購入で簡単にグレードアップが可能です。
- 当然、2M/4M/6Mモデルでは購入後も8Mまでのメモリ増設が可能です。
- XVI用の専用内蔵RAMとの共存はできません。

製品	型番	価格
2Mメモリ数値演算プロセッサ無し	PRKII-02	¥ 55,000
4Mメモリ数値演算プロセッサ無し	PRKII-04	¥ 90,000
6Mメモリ数値演算プロセッサ無し	PRKII-06	¥ 125,000
8Mメモリ数値演算プロセッサ無し	PRKII-08	¥ 160,000
2Mメモリ数値演算プロセッサ付属	PRKII-12	¥ 85,000
4Mメモリ数値演算プロセッサ付属	PRKII-14	¥ 120,000
6Mメモリ数値演算プロセッサ付属	PRKII-16	¥ 155,000
8Mメモリ数値演算プロセッサ付属	PRKII-18	¥ 190,000

BASIC HOUSE 計測技研

お申し込み・お問い合わせは **0286-22-9811**(代)

〒321 栃木県宇都宮市竹林町503-1 FAX 0286-25-3970

株式会社 デンキヤ

〒332 埼玉県川口市西川口4丁目6番4号
AM11:00〜PM7:00 水・木定休

今月の超特価品

シャープ
X68000セット
XVI

特価 ［　　　　　　］
TEL

★X6800本体★	
CZ-644C-TN	¥
CZ-634C-TN	¥
CZ-653C	¥ 192,400
CZ-623C-TN	¥ 323,700
CZ-604C-TN	¥ 226,200
★X6800ディスプレイ★	
CZ-606D	¥ 53,900
CZ-613D	¥ 91,100
CZ-605D	¥ 77,600
CZ-604D	¥ 64,000
CU-21HD	¥ 99,900
★プリンタ・ケーブル付★	
CZ-8PG1	¥ 90,400
CZ-8PG2	¥ 111,200
CZ-8PK10	¥
CZ-8PC5	¥ 67,300
IO-735X	¥
CZ-6PV1	¥
★RAMボード★	
CZ-6BE1B	¥ 21,000
CZ-6BE2	¥
CZ-6BE4	¥
PIO-6BE1-A	¥ 18,100
PIO-6BE2	¥ 33,800
PIO-6BE4	¥ 59,400
CZ-6BE2A	¥ 44,900
CZ-6BE2B	¥ 41,000
★その他★	
CZ-6BP1	¥
CZ-6EB1	¥

★ハードディスク各種★	
CZ-620H	¥
CZ-64H	¥ 90,000
IT X80	¥
IT X130	¥
HXD140	¥
HXD040	¥
HXD042	¥
AV-090WS	¥ 116,800
AV-050WS	¥ 93,100
★インターフェイス各種★	
CZ-6BS1	¥ 22,400
CZ-6BM1	¥ 20,100
CZ-6BV1	¥ 15,800
CZ-6BF1	¥
CZ-6BG1	¥
CZ-6BU1	¥
CZ-6BC1	¥
CZ-6BL1	¥
CZ-6BL2	¥
CZ-6BP2	¥
★周辺機器各種★	
CZ-8NJ2	¥ 17,900
CZ-8NJ1	¥ 1,300
CZ-8NM3	¥ 7,400
CZ-8NT1	¥ 10,400
CZ-8NM2A	¥ 5,100
BF-68PRO	¥ 13,800
CZ-6TU-BK	¥ 23,000
CZ-6VT1	¥ 48,500
CZ-6SD1	¥

★ソフト各種★	
CZ-249GS	¥ 22,400
CZ-255GS	¥ 6,600
CZ-256GS	¥ 6,600
CZ-245LS	¥ 33,600
CZ-260LS	¥ 7,400
CZ-251BS	¥ 29,900
CZ-243BS	¥ 14,900
CZ-240BS	¥ 11,100
CZ-259SS	¥ 5,100
CZ-257CS	¥ 14,900
CZ-219SS	¥ 22,400
CZ-252MS	¥ 21,600
CZ-213MS	¥ 14,100
CZ-247MS	¥ 21,600
★モデム各種★	
MD24FB5V	¥
MD24FS7	¥
MD24FP5II	¥
PV-M24B5	¥
PV-A24B5	¥
コムスターズ2424/5	¥ 27,800
コムスターズ2424/4	¥
SR-120S	¥
SR-240S	¥
SR-240V	¥
★ゲームソフト各種★	

24時間テレホンサービス
0482-54-3444

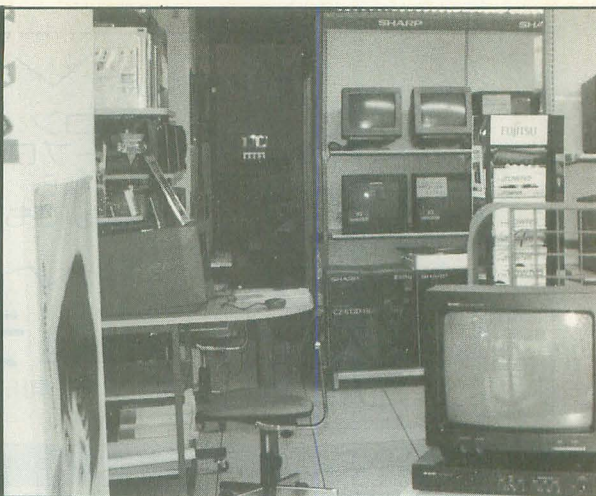
お申し込みはお電話で
TEL 0482-54-3400
FAX 0482-54-3443

★振込先★
三菱銀行西川口支店
普通0258081
㈱デンキヤ

X68000

XVIサマーフェアー!!

大特価セール実施中!

X68000下取りセール実施中!　'91年8月15日迄

X68000　X68000セットで

XVI
CZ-634CTN
+CZ-606DTN
フェアー特価

XVI HD
CZ-644CTN
+CZ-606DTN
フェアー特価

EXPERT
CZ-602CGY
+CZ-603DGY ¥248,000
+CZ-612DGY ¥268,000

XVI
CZ-634CTN
+CZ-614DTN
フェアー特価

XVI HD
CZ-644CTN
+CZ-607DTN
フェアー特価

EXPERTII
CZ-603C
+CZ-606D ¥270,000
+CZ-607D ¥285,000
+CZ-614D ¥310,000

XVI
CZ-634CTN
+CZ-607DTN
フェアー特価

XVI HD
CZ-644CTN
+CZ-613DTN
フェアー特価

Super
CZ-604CTN
+CZ-606DTN ¥290,000
+CZ-607DTN ¥305,000
+CZ-614DTN ¥330,000

X68000周辺機器

14型カラーディスプレイ
CZ-606D-BK
標準価格　¥79,800
特価

ドットマトリクスカラー漢字プリンタ(80桁)
CZ-8PG1
標準価格　¥130,000
特価 **¥104,000**

ドットマトリクス漢字プリンタ(136桁)
CZ-8PK10
標準価格　¥97,800
特価

カラーイメージユニット
CZ-6VT1
標準価格　¥69,800
特価 **¥55,800**

カラービデオプリンタ
CZ-6PV1
標準価格　¥198,000
特価 **¥158,000**

ドットマトリクスカラー漢字プリンタ(136桁)
CZ-8PG2
標準価格　¥160,000
特価 **¥128,000**

24ピン漢字プリンタ(80桁)
CZ-8PK7
標準価格　¥122,000
特価

光磁気ディスクユニット
CZ-6M01
標準価格　¥450,000
特価 **¥360,000**

X68000ソフト

		正価	特価
ICONEDITER	計測技研	¥ 4,800	¥ 4,100
C言語ライブラリー(X68000)	計測技研	¥ 6,800	¥ 5,800
BASIC拡張関数パッケージ	計測技研	¥ 9,800	¥ 8,500
DISK CACHER	計測技研	¥ 6,800	¥ 5,800
たーみのる2	SPS	¥17,800	¥14,250
X1エミュレータ	ACCESS	¥ 9,800	¥ 8,330
EM 68K(エミュレータ)	ニューウェーブ	¥30,000	¥25,500
CP/M 68K	ニューウェーブ	¥110,000	¥93,500
サイクロン Ver1.2	アンス	¥58,000	¥49,300

		正価	特価
SUPER DIVICE MONITOR "T"	ブルースカイ	¥15,000	¥12,000
CANVAS	CZ-249GS	¥29,800	¥23,850
XBAS to C CHECKER	CZ-260LS	¥ 9,800	¥ 7,850
Multiword	CZ-225BS	¥32,000	¥25,600
Human68K Ver2.0	CZ-244SS	¥ 9,800	¥ 7,850
C compiler Ver2.0	CZ-245LS	¥44,800	¥35,850
SOUND PRO68K	CZ-214BS	¥15,800	¥12,650
Sampling PRO68K	CZ-215MS	¥17,800	¥14,250
MUSIC PRO68K	CZ-213MS	¥18,800	¥15,000

		正価	特価
MUSIC Studio PRO68K Ver2.0	CZ-261MS	¥28,800	¥23,000
MUSIC PRO68K MIDI	CZ-249MS	¥28,800	¥23,000
CARD PRO68K Ver2.0	CZ-253BS	¥29,800	¥23,850
CARD活用フォーム集1	CZ-242BS	¥ 9,800	¥ 7,850
SX-WINDOW Ver1.1	CZ-278SS	¥ 9,800	¥ 7,850
OS-9	CZ-219SS	¥29,800	¥23,850
Easypaint SX-68K	CZ-263GW	¥12,800	¥10,250
Teleportion PRO68K	CZ-258BS	¥22,800	¥18,250

お買上げの方に ソフト3本(当店指定)+フロッピーシール 2点セット プレゼント(8/15まで)

EXPERT+HD
CZ-602CGY+HXD140
+CZ-603DGY **¥315,000**
+CZ-612DGY **¥335,000**
+CZ-613DGY **¥355,000**

カラープリンタセット
CZ-634CTN
+CZ-607DTN
+MZ-1P22
+ケーブル **¥360,000**

EXPERTフルセット
CZ-602CGY
+HXD140
+CZ-8PK10 **¥375,000**

Super HD
CZ-634CTN
+CZ-606DTN **¥380,000**
+CZ-607DTN **¥395,000**
+CZ-614DTN **¥420,000**

PROIIフルセット
CZ-653CBK
+CZ-606DBK
+HXD140
+MZ-1P22
+ケーブル **¥328,000**

Superカラーセット
CZ-634CTN
+CZ-607DTN
+IO-735XB
+JX-220X **¥707,000**

PROII
CZ-653CBK
+CZ-606DBK
(のこり少々)
大特価

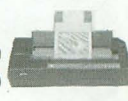
Cコンパイラセット
CZ-644CTN
+CZ-613DTN
+CZ-8PK7
+CZ-245LS **¥566,000**

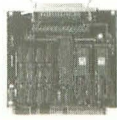
EXPERTIIテレポーテーションセット
CZ-603C
+CZ-606D
+CZ-8PG2
+CZ-8TM1
+CZ-258BS **¥418,000**

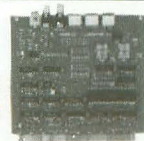
HXD 040 23ms X68000
標準価格¥118,000 **特価¥79,800**
HXD 042 X68000 増設用
標準価格¥128,000 **特価¥102,500**
HXD 140 X68000 内蔵用
標準価格¥98,000 **特価¥75,000**
HXD 140は602C, 603C, 652C, 653Cの内蔵用

1MB増設RAMボード(内蔵用)
CZ-6BE1
標準価格¥35,000
特価¥29,800
CZ-6BE1B
標準価格¥28,000
特価¥19,500

X68000 3.5インチフロッピーディスクユニット
X6835-2F
標準価格 ¥80,000
特価

カラーイメージジェットプリンタ
IO-735X-B
標準価格 ¥248,000
特価 ¥198,500

RGBシステムチューナー
CZ-6TU-BK
標準価格 ¥33,100
特価 ¥26,000

SCSIボード
CZ-6BS1
標準価格 ¥29,800
特価 ¥19,500

カラーイメージスキャナ
232Cケーブル, スキャナツールソフト付
JX-220X
標準価格¥168,000
特価¥134,500

MIDIボード
CZ-6BM1
標準価格 ¥26,800
特価 ¥21,400

※富士通、NEC、シャープ周辺機器(拡張機器全機種、プリンター他)も常時取り扱っております。

〈全商品新品完全保証付〉 シャープ、カシオポケコン全機種取扱い。カタログ、価格表ご請求には、72円を添えてお願い致します。

0426-45-3002(京王線 北野駅前店)-3001(本 店) -3003(教 室)
FAX.0426-44-6002
●営業時間/10:00~19:00●電話受付/20:00迄可●定休日/水曜日
SHARP SUPER XEX SHOP
アイビット電子株式会社 〒192 東京都八王子市北野町560-5

上記の広告商品はすべて店頭販売もしております。

全国通信販売
北海道から沖縄まで
★送料はご注文の際にお問い合わせ下さい。
★掲載の商品は、すべて新品、保証書付きです。
★掲載の商品は充分用意してありますが、ご注文の際は、在庫の確認の上、現金書留または、銀行振込でお申し込み下さい。全商品クレジットでも扱っております。
★お申し込みの際は必ず電話番号を明記して下さい。
★商品、品切れの節はご容赦下さい。

富士銀行八王子支店 (普)1752505

●本誌発売時には上記価格よりさらにお求めやすい価格に変更されている場合があります。●この広告の商品にはすべて送料・消費税は含まれておりません。

名古屋 大須店 開店記念

7月19日 OPEN!!

大須店 ☎(052)265-1650(代)
〒460 名古屋市中区大須3－11－19 OAビル

X68000シリーズ 全店特価にてご奉仕！

XVIセット
CZ-634C-TN
＋
CZ-606D-TN
定価合計 ¥447,800
大特価!!

XVIセット
CZ-634C-TN
＋
CZ-613D-TN
定価合計 ¥503,000
大特価!!

XVI-HDセット
CZ-644C-TN
＋
CZ-606D-TN
定価合計 ¥597,800
大特価!!

XVI-HDセット
CZ-644C-TN
＋
CZ-613D-TN
定価合計 ¥653,000
大特価!!

SUPERセット
CZ-604C-TN
＋
CZ-606D-TN
定価合計 ¥427,800
大特価!!

SUPER HDセット
CZ-623C-TN
＋
CZ-613D-TN
定価合計 ¥633,000
大特価!!

EXPERTⅡセット
CZ-603C-BK
＋
CZ-606D-BK
定価合計 ¥417,800
大特価!!

PROⅡセット
CZ-653C-BK
＋
CZ-606D-BK
定価合計 ¥364,800
大特価!!

IO-735X(B)
48ドットカラーインク
ジェットプリンター
標準小売価格 ¥248,000
大特価!!

JX-220(B)
フルカラーイメージスキャナ
X68K用 パラレルI/Fボード
標準小売価格 ¥168,000
大特価!!

HAL研究所
ファインスキャナー
68K(HGS-68)
標準小売価格 ¥39,800
¥31,800

MIDI関連
音源モジュール＋互換MIDI I/Fセット

ローランド CM-64
システムサコム SX-68M
定価合計 ¥148,800
大特価!!

ローランド CM-32L
システムサコム SX-68M
定価合計 ¥88,800
大特価!!

増設ラムボード	「XVI」専用増設ボード
PIO-6BE1-A (1MB) ………¥18,000	CZ-6BE2A (2MB) XVI本体内蔵メモリ ……………¥46,000
PIO-6BE2-2M (2MB) ………¥34,800	CZ-6BE2B (2MB) メモリーボード増設用 ……………¥42,800
PIO-6BE4-4M (4MB) ………¥60,000	CZ-6BP2 (数値演算プロセッサー) ……………¥35,800

直接ご来店頂けない場合は、
通信販売もご利用いただけます。
お電話でお申し込みください。
☎(052)332-5688

銀行振込
各店舗に御予約、ご注文いただきましたら、最寄の銀行から当社指定銀行口座へ電信扱にてお振り込み下さい。手数料はお客様負担になります。

代金引き替え配送
お電話で商品の注文が出来ますお客様宅へ配達時、商品と引き替えにお代金をお支払いいただきます。商品代金の他に手数料かかかります。

クレジット
お電話にてお申込みいただきましたら折り返し弊社より専用申込用紙をお送りいたします。必要事項記入の上ご返送下さい

いずれも商品在庫をご確認の上お申し込みください。

※表示価格には消費税は含まれておりません。

ITEC製 新型ハードディスク

ブラックボディ

X68000専用
80MB SCSI/SASI兼用ハードディスク
TX-80 アクセスタイム20ms
標準小売価格 ¥108,000
→**¥88,000**

グレーボディ

130MB SCSI ハードディスク
TX-130 アクセスタイム20ms
標準小売価格 ¥138,000
→**¥110,000**

180MB SCSI ハードディスク
TX-180 アクセスタイム20ms
標準小売価格 ¥185,000
→**¥148,000**

その他、各種周辺機器、中古品等
多数取り扱っております!!
お近くの「OAシステムプラザ」へ
まず、お電話を!!
お待ちしております。

TREASURE BOX.

店舗	電話	店舗	電話
札幌店	011-210-8812	メガタウン店	052-242-8433
仙台店	022-268-5541	京都店	075-344-0347
東京店	03-3255-9188	大阪店	06-632-4233
横浜店	045-314-6634	大阪日本橋店	06-646-3169
浜松店	053-458-3755	岡山店	0862-21-4133
名古屋店	052-332-5233	広島店	082-240-9669
名古屋アメ横店	052-264-9715	福岡店	092-714-0030
アメ横2階店	052-262-6909	福岡ユーテク店	092-733-8931

X68000 PROSHOP
㈱OAシステムプラザ
本社 愛知県名古屋市中区大井町3-20 OAビル

ACCESS

X1エミュレータ

好評発売中

定価¥9,800

X1エミュレータはX68000上でX1シリーズのアプリケーションを実行するためのソフトエミュレータです。X1のアプリケーションを完全にソフトウェアのみでエミュレートしているため、X1上での実行速度と比較して、平均3～5倍程度おそくなりますが、X68000のマシン上に実現した仮想X1マシンを楽しめます。また、X1とX68000の相互間でファイルを転送するためのユーティリティと専用ケーブルが付属しますので、X1上で作り上げたソフトの資産をX68000上に移行することも簡単にできます。

X1エミュレータの機能

- X1エミュレータはX1に相当する機能をエミュレート。この仮想コンピュータには最大4つのドライブが仮想的に接続。
- X1エミュレータからみたドライブはHuman68kのドライブ上にあるファイルで仮想的に実現。このファイルはX1用の5" 2Dディスクのイメージをファイル転送ユーティリティでまるごと転送したもの。
- X1エミュレータで仮想的に実現したX1は仮想ドライブから起動。このため仮想ドライブ用ファイルには、X1を立ち上げるために必要なHuBASICやCP/Mなどのシステムプログラムが必要。
- X1エミュレータでは、X1の持つVRAMを含むメモリイメージやZ80CPUを仮想的にソフトウェアで実現。

ファイル転送ユーティリティ

ディスク転送

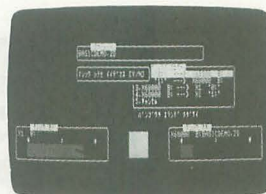
X1ディスク⟷X68000 Human68k(5" 2Dディスクイメージファイル)

- X1エミュレータではHuman68k上のディスクイメージファイルを仮想ドライブとして使用。

ファイル転送

X1 BASIC : CP/M⟷X68000 Human68k

- X1で作ったプログラム&データをX68000上で使用。

※付属の専用ケーブルをX1とX68000に接続してファイルを転送します。

X1エミュレータ Q&A

Q. ファイル転送のために別途RS-232Cケーブルを買わないといけないのですか?

A. 専用のケーブルが付属しますのでその必要はありません。

Q. X1BASICのプログラムをX68000上のX-BASICで使えますか?

A. 通常のセーブではコードが違うので使用できませんが、アスキーセーブしたファイルであればX-BASIC上でそのままロード可能です。

Q. TurboBASICで作成した住所録などの漢字を含んだデータがあるのですがX68000上にファイル転送できますか?

A. X1TurboもX68000も漢字はシフトJISコードなのでファイルの転送は可能です。ただし、漢字ROMを必要とするものはサポートしていません。

Q. Turbo用のソフトは動きますか?

A. X1用のみでTurbo専用のソフトは動きません。

Q. ゲームは動きますか?

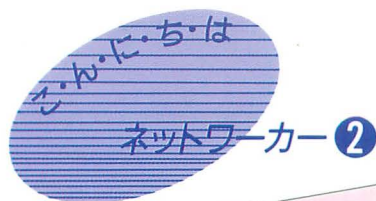
A. 純粋にBASICでかかれたものは動きますが、プロテクトがかかったものや直接ハードをアクセスするような市販のゲームは動きません。

＊タイミング等ハードウェアに依存するようなソフトは、原理上実行できない、もしくは正常に動作しない場合がありますのでご注意ください。
＊一部サポートしていない機能があります。

X1エミュレータ通信販売 購入希望として住所、氏名、電話番号をお知らせください。注文書をお送り致します。

＊この商品価格には消費税は含まれておりません。
＊CP/Mはデジタルリサーチ社の商標です。
文中のソフトウェアは各社の商標です。
＊製品の仕様、名称は予告なく変更する場合もございますのであらかじめご了承ください。

有限会社アクセス 〒101 東京都千代田区神田神保町1-64 神保町協和ビル7F
☎03(3233)0200(代) FAX.03(3291)7019

資料請求券 oh!X 8月号

パソコン/ワープロ通信ネットワークサービス

J&P HOT LINE

J&P HOTLINEは
私の頼もしいスタッフの一人。

原　陽子さん　23歳

(JH330341はらよ) 企画・広告デザイナー

チャレンジ精神旺盛な原さんは、新入社員時代から企画書や広告のラフ案作成にパソコンを活用するべく試行錯誤。今では、デザインルームのチーフとしてパソコンとワープロを使い分けながらてきぱきと仕事をこなしておられます。「デザイナーは、パソコンを活用はしても頼りすぎてはだめ」と、きっぱり言い切った一言が印象的でした。

NAPLPS画像通信に大きな期待。ただいま、作品制作中!

J&P HOTLINEが実験的にスタートさせる新ホストの案内で「NAPLPS画像通信」という見慣れないことばに出会ったとき、「画像」の2文字が大きく目に飛び込んで来たという原さん。

持ち前の行動力で事務局に問い合わせ。今までいろんなSIGのJISグラフィックコーナーや画像通信コーナーを見たり聞いたりしても、もう一つピンと来なかったのが、事務局から説明を受けて「これは、おもしろそう」と大いに期待を持ったとのこと。さっそくお絵描きソフトと画像通信用の通信ソフトを入手して、色々と作品を制作中。

デザイナーという職業柄、お絵描きソフトは「お遊び」という先入観を持っていたことを今では反省。仕事に活かすだけでなく、趣味としても十分に魅力的で、アニメーションやディスプレイ上の絵本制作にまで夢は広がります。近々、力作をひっさげてNAPLPS画像通信にデビュー予定とのことなのでお楽しみに!!

原さんとJ&P HOTLINEのお付き合いは、
もちろん画像通信だけではありません。

- ★ 宝くじマニア(?!)なのに、買うだけ買って発売日を忘れてしまう…宝くじ当選番号情報は力強い味方です。
- ★ 仕事が忙しいからこそ充実した自分の時間を持ちたい……データベースのCD情報と文庫本の新刊情報は定期的に欠かさずチェック!
- ★ 勤務先が心身症に関する財団のお仕事のお手伝いもしているので、関連するSIGのHeartful VoiceやサイコロジストもできるだけこまめにROM。
- ★ 企画の仕事が結構多いので、時間があれば不特定のSIGやBBSを見て回っています。ただ、なかなか特定のSIGに参加する時間がないので、どうしてもROMっ子になってしまい、書き込めないのが悩みとのことでした。

J&P HOTLINEへのご入会はスタータキットで。

お求めは、下記のお店へ。又は現金書留にて、¥3,000+¥90(消費税3%)=¥3,090を事務局までお送り下さい。
すぐにスタータキットをお送りします。

お問い合わせは　〒556 大阪市浪速区日本橋西1-6-5 上新電機株式会社
J&P HOTLINE事務局宛　TEL.(06)632-2521

スタータキットのお求めはJ&P各店でどうぞ。

店名	住所	電話
渋谷店	東京都渋谷区道玄坂2丁目28番4号	☎(03)3496-4141
町田店	東京都町田市森野1丁目39番16号	☎(0427)23-1313
八王子店	東京都八王子市旭町1番1号八王子そごう7F	☎(0426)26-4141
立川店	東京都立川市幸町4－39－1	☎(0425)36-4141
本厚木店	厚木市中町3－4－3	☎(0462)25-1548
富山店	富山市桜町2－1－10	☎(0764)32-3133
金沢店	金沢市入江2－63	☎(0762)91-1130
寺地店	金沢市寺地2－3	☎(0762)47-2524
大須店	名古屋市中区大須4丁目2-48	☎(052)262-1141
テクノランド	大阪市浪速区日本橋5丁目6番7号	☎(06) 634-1211
メディアランド	大阪市浪速区日本橋5丁目8番26号	☎(06) 634-1511
㊊コスモランド	大阪市浪速区難波中2丁目1番17号	☎(06) 634-3111
U. S. LAND	大阪市浪速区日本橋4丁目9番15号	☎(06) 634-1411
ビジネスランド	大阪市北区梅田1-1-3大阪駅前第3ビルB2	☎(06) 348-1881
梅田店	大阪市北区小松原町1－10	☎(06) 362-1141
高槻店	高槻市高槻町11番16号	☎(0726)85-1212
くずは店	枚方市楠葉花園町15番2号	☎(0720)56-8181
千里中央店	豊中市新千里東町1-3 SENCHU PAL 2番街4F	☎(06) 834-4141
摂津富田店	高槻市大畑町24－10	☎(0726)93-7521
寝屋川店	寝屋川市緑町4－20	☎(0720)34-1166
藤井寺店	藤井寺市岡2丁目1番33号	☎(0729)38-2111
岸和田店	岸和田市土生町2451－3	☎(0724)37-1021
さんのみや1ばん館	神戸市中央区八幡通3-2-16	☎(078)231-2111
西宮店	兵庫県西宮市河原町5－11	☎(0798)71-1171
姫路店	姫路市東延末1丁目1番住友生命姫路南ビル1F	☎(0792)22-1221
京都寺町店	京都市下京区寺町通仏光寺下ル恵比須之町549	☎(075)341-3571
京都近鉄店	京都市下京区烏丸通七条下ル東塩小路町702	☎(075)341-5769
和歌山店	和歌山市元寺町4丁目4番地	☎(0734)28-1441
奈良1ばん館	奈良市三条町478－1	☎(0742)27-1111
郡山インター店	大和郡山市横田693－1	☎(07435)9-2221
熊本店	熊本市手取本町4－12	☎(096)359-7800

T4910217908609 雑誌02179-8